

Commodore

WORLD

Nº 2. MARZO 1984

275 PTAS.

VIAJES
a
ESTADOS UNIDOS
Para nuestros lectores
Página 4

El "duende" de los Commodore

Rincón del 700

Añade trece comandos a tu Vic

Club Commodore

Conversión de programas Vic a C-64

Juegos:

Disfruta las matemáticas

y...





GALERIA DE SOFT



El programa para el procesador de textos para el Commodore-64. EASY SCRIPT convierte al C-64 en un procesador de textos muy potente que dispone de todas las facilidades de una máquina dedicada que pudiera costar miles de libras.

Está escrito totalmente en lenguaje máquina, el cual permite una velocidad realmente alta, con una capacidad de aproximadamente 30.000 caracteres de texto, es decir, más de 750 líneas de texto. Con EASY SCRIPT los textos se pueden crear, salvar, recuperar y modificar mediante el uso bien, de la unidad de disco 1541 o la grabadora C2N. Incluso se puede transferir la información tanto de la unidad de disco a la grabadora como de la grabadora a la unidad de disco.

El texto se almacena en disco o en cinta en unos ficheros de datos estándar que pueden ser leídos por los programas en Basic. De esta forma EASY SCRIPT dispone de la capacidad de leer la información creada a partir de un programa en Basic.



DE COMPRAS

PRECIO: 20.000 PTS.

Para introducir un texto muchas teclas funcionan igual que para un programa en Basic, por ejemplo las teclas del control del cursor.

Además, 55 potentes comandos de editar están disponibles al pulsar una tecla.

Una vez introducido el texto, existen más de 35 comandos de formateado que pueden ser utilizados para modificar la forma en que se imprime el texto.

Algunas de las facilidades disponibles en EASY SCRIPT se presentan a continuación:

- Búsqueda y Sustitución.
- Relleno automático de bloques variables.
- Centrado y Justificación.
- Modalidades de Borrado e Inserción.
- Desplazamiento (Scrolling) en todas las direcciones.
- Serie de ficheros encadenados.
- Tabulaciones horizontales y verticales.
- Modalidad y encolumnado de números decimales.
- Cabeceras y Pies.

EASY SCRIPT

Procesador de Textos para C-64

EASY SCRIPT no solamente funciona con las impresoras matriciales 1515 y 1525, sino que también maneja otras marcas populares, incluyendo la serie EPSON MX y hasta las impresoras de

margarita con letras de calidad de imprenta.

El texto se introduce fácilmente en EASY SCRIPT y sólo hace falta pulsar la tecla RETURN al final de cada párrafo.

Numeración automática de las páginas.
Manejo de discos y cintas.
Combinación (merge) de párrafos o ficheros.
Pleno control de la impresora.

En el próximo número se publicará un amplio artículo sobre cómo generar Caracteres Especiales con EASY SCRIPT

LAS TORRES DE HANOI

Para el VIC-20

PRECIO: 1.500 PTS.

Tradicionalmente este juego se hace con tres palos verticales y unos aros de diámetros diferentes. Al inicio se colocan todos los aros en un mismo palo de menor a mayor. El juego consiste en pasarlos a otro palo. Solo hay dos reglas: Nunca puede estar un aro sobre otro de diámetro menor, y sólo se puede tomar el aro superior de cada torre.

Estos paquetes están distribuidos por Microelectrónica y Control

MATEMATICAS-1

PROGRAMA
NIVEL BUP
PARA VIC-20 CON
8 ó 16 K

PRECIO: 2.000 PTS.
en cinta

Este es un programa que plantea tests de matemáticas y geometría. El VIC preguntará el tema que prefiere, Usted cargará la cassette de datos, una vez el VIC haya encontrado el cuestionario, usted podrá empezar a contestar. Usted puede pasar pulsando "Return" o finalizar el test tecleando "Fin". Una vez finalizado el test, se le presentará la puntuación, el tiempo empleado y la posibilidad de contestar las preguntas pasadas. Si dice "N", el VIC ofrecerá un menú con tres posibilidades:

- Otro test
- Revisar test
- Fin

En cuyo caso el VIC presentará la portada del programa.

NOTA: En el test de tiempo no dice si la contestación es correcta o no.

CONTENIDO	PAG.
EDITORIAL	4
CLAVE PARA INTERPRETAR LOS LISTADOS	6
DIVIERTETE CON LAS MATEMATICAS Para repasar las matemáticas estudiadas en el colegio	7
DISFRUTA MUCHO MAS CON LA MAGIA DE LA CONVERSION Cómo convertir los numerosos programas del VIC para que se procesen en el C-64	10
VENTANA CBM Algunos aspectos del sistema co-residente MEC/DOS	18
¡ARRIBA PERISCOPIO! El programa periscopio proporciona un instrumento útil para que entendamos mejor nuestras máquinas	22
DISK-O-VIC El Commodore VIC-20 y la unidad de discos 1541 constituyen una combinación muy importante	26
BATALLA NAVAL Juego	37
CLUB COMMODORE	40
Topo Loco	41
Funy Faces	42
Carta Blanca	43
Magia	46
Rincón del 700 Software para el 700. El aspecto más importante del 700 en lo que a software se refiere es MEC/DOS	50
Rincón del Vicolage El RS-232-C en los ordenadores Commodore	52
MAS ALLA DEL MANUAL Un truco que cada programador debe tener en su repertorio	55
FICHEROS EN DISCO Indexados Secuenciales	57
GALERIA DE SOFT	2 y 59

PROXIMO NUMERO

- Curso lenguaje Máquina
- Exprimiendo el jugo a las teclas de funciones
- Base de datos para el VIC 20 y el C-64
- Caracteres especiales con Easy Script
- Atajo hacia el color

- Chavales... ¡Sorpresa!

- Recetario de Pokes

- Juegos

- Club Commodore

y todas vuestras colaboraciones

¡OS ESPERAMOS!

Commodore World es publicado en colaboración entre Microelectrónica y Control-Commodore y SIMSA

EQUIPO

Manuel AMADO; Adela LOPEZ; Maria LOPEZ; Juan MARTINEZ;
Pere MASATS; Jeffrey MILLS; Rafael NAVARRO; Fernando M. RODRIGUEZ;
Diego ROMERO; Albert SANGLAS; Manuel SANS; Jordi SASTRE; Valerie SHANKS...
... Y NUESTROS LECTORES

SIMSA

Coordinador María López
Pedro Muguruza, 4-8ºB — Madrid-16 — Tlf.: (91) 259 54 78
Sant Gervasi de Cassoles, 39 despacho 4
Barcelona-22 — Tlf.: (93) 212 73 45

MICROELECTRONICA Y CONTROL-COMMODORE

Coordinador Pere Masats
Taquígrafo Serra, 7-5º
Barcelona 29
Teléfono (93) 250 51 03/02

SUPER-INTERESANTISIMO

Un lector... (¿o más...?) viajará a Estados Unidos con Commodore World

Porque estábamos seguros del éxito de Commodore World, pero no tanto, sabíamos que iba a "caer bien" y sobre todo que iba a llenar un hueco muy importante en la gran familia Commodore —sin embargo, sinceramente, no nos esperábamos la súper gran acogida que ha tenido—. Hoy al redactar estas líneas, 15 de febrero, a sólo 15 días de su salida a la calle, hemos superado la línea de los 3.500 suscriptores y nos estamos acercando rápidamente a los 4.000 y el buzón se nos ha quedado pequeño recibiendo correspondencia de todos los que formáis el gran equipo de Commodore World. Y... se acabó el hablar y darnos auto-bombo, vamos al grano y seguid leyendo, que lo que sigue es de "super-interés" para todos.

Suscriptores y el nº 5.000... ¡BINGO!

Las suscripciones se acumulan en la redacción y en el teléfono. En el momento en que se abre un sobre, se recibe una tarjeta o se apunta una suscripción telefónica, se apunta el número correspondiente a esa suscripción... Pues bien, **el suscriptor al que le corresponda el nº 5.000, y ya hemos superado los 3.500, se encontrará automáticamente con un viaje a Estados Unidos, donde visitará la fábrica y casa madre de Commodore...** Sí, habéis leído bien... no es ningún error de imprenta. ¡Bueno! ¿Y los demás suscriptores, qué? —Eso no es justo— vale, de acuerdo, sobre todo para los 4.999 primeros que han hecho posible que el número 5.000 cante BINGO. Pues, aunque no está decidido todavía qué será, adelantaremos que organizaremos algún tinglado para todos, tanto los pre-5.000 como los post-5.000 y **algo especial** para los primeros 4.999 que anunciaremos en cuanto se haya decidido, y veréis que nosotros no nos dormimos en los laureles y hacemos las cosas rápido (adelantamos que hay viajes y aventuras por medio).

Los "VIEJOS LEALES" y los nº 13, 14 y 15

Hemos tenido algunas llamadas de antiguos suscriptores del Club Commodore que no han recibido los nº 13, 14 y 15. Lo más probable es que por coincidir con la época de transición haya habido algún traspapeleo involuntario. A los que os haya

ocurrido esto, **escribid** (no por teléfono) a Microelectrónica y Control de Barcelona, c/ Taquígrafo Serra, 7, especificando a la atención de Pere Masats. Pere, tan imprescindible hoy en Commodore World como lo fue antes en Club Commodore, estará encantado de remitirlos los ejemplares que os faltan a vuelta de correo.

Market-Club

Mercadillo-Clubs-Bolsa de trabajo

Pretendemos que MARKET-CLUB sea una importante sección al servicio de toda la familia Commodore. A partir de hoy mismo se dividirá en tres apartados: MERCADILLO, CLUBS y, muy importante, BOLSA DE TRABAJO.

MERCADILLO: Será para todo el cambalache, equipos usados, ofertas, demandas, etc. **CLUBS:** La propia palabra lo explica. Información de Club Commodore por todo el país tanto de los que existen como de usuarios que están buscando uno ya formado en su zona o de aquellos que quieran formar uno nuevo.

BOLSA DE TRABAJO: Este es el apartado que con mayor ilusión comenzamos y pretendemos que sea una importantísima "estrella" de Commodore World. Por un lado, un gran número de "Commodorianos" son gente joven que está buscando empleo por primera vez, por otro lado, dentro de los "Commodorianos" maduros y semi-maduros hay los que se encuentran en paro y creemos que nuestra revista puede prestar un gran servicio en este campo.

Todos los anuncios serán gratuitos excepto para las Compañías que ofrecen servicios, productos o empleo a quienes se les cobrará 300 pesetas por línea.

Todos los anuncios llevarán un número de referencia. Os rogamos que cuando ya no se necesite publicar, nos lo comuniquéis, nombrando este número, para suprimirlo.

Programas y errores...

A partir de este número que tenéis en la mano, podemos garantizaros que todos los programas y juegos que se incluyen, vayan completamente comprobados y sin el más mínimo error (dentro de lo humanamente posible). Diego, el "cerebro" del equipo editorial ha hecho todas las horas extraordinarias del mundo para asegurarse de que todos los listados van perfectos.

Sin embargo, lamentamos que en nuestro primer número, debido al gran trabajo de lanzamiento, no pudimos ser tan concienzudos y parece ser que se nos introdujo un duende travieso en la línea 850 de Video Casino-Tiro al Blanco. Un lector, nos acaba de llamar diciendo que tenía problemas con esta línea. Vamos a revisar este programa y os diremos lo que hay en nuestro próximo número. Quisiéramos poder sacaros de dudas ahora mismo pero tenemos que entrar en imprenta y no da tiempo.

...Y servicio de cintas para los más vagos.

Aquellos lectores que os canséis de teclear un programa (no os preocupéis ni os dé vergüenza, ¡sucede en todo el mundo!) (¡por eso existe el soft!) podéis pedir a la editorial la cinta del programa y número de Commodore World en que aparece, adjuntando un cheque por el importe: 875 pesetas por cinta y 75 pesetas para gastos de envío —cuidado, 75 pesetas por paquete, no por cinta—. Los que vivís en Madrid podéis recoger las cintas en la editorial y no pagar las 75 pesetas (llamad antes por teléfono para tenerlas preparadas).

Carta Blanca y Seamos Preguntones

Dos nuevas secciones: CARTA BLANCA habla por sí misma —cartas del lector con todo lo que tengáis que decir, opinar o sugerir— tenéis "carta blanca". SEAMOS PREGUNTONES —seámoslo, por favor— preguntad todo lo que os interese sobre los Commodore, su funcionamiento, formas de darle mayor rendimiento o capacidad, programación, etc. Contestaremos todo lo que podamos y si hay algo que no sepamos contestar, la pregunta queda hecha como desafío a los lectores "listos".

Chavales Especial para vosotros

No nos hemos olvidado de vosotros. Aunque sabemos que los "Commodorianos" infantiles son unos superlistos que pueden manejar un montón de las cosas que aparecen en la revista, a partir del próximo número tendréis la sección especial para los

más jovencillos de la familia, aunque mucho nos tememos que los adultos la van a echar más de un vistazo a hurtadillas. Así que atención, nuestros "duendes" ROM y RAM os esperan en el próximo número y con algún programilla especial para vosotros —que no pase como en "el tren" que luego es papá el primero que lo usa... (¡o mamá!).

Curiosidades sobre los Commodore

El otro día nos dijeron que un famoso campanario de un pueblo está funcionando de nuevo después de muchos años de estar casi callado. El milagro se debe a que esta siendo manejado por un VIC y unos dicen que consigue un cambio de melodías precioso. Los que sepáis de casos en que se emplean los Commodore para cosas que os sorprendan, contádnoslo —puede dar ideas a la gente de lo mucho que pueden hacer con una "maquinilla" y al mismo tiempo puede haber anécdotas divertidas.

Colaboraciones

Recordamos que para todas las colaboraciones que se envíen va a haber sorteos y premios (ver Commodore World nº 1). Como los premios especiales a los mejores

van a darse por grupos de edad, no olvidéis poner vuestra edad. En nuestro número 1 decíamos, por error de imprenta, que los programas deberían venir todos con cinta o disco —rectificamos— deben venir todos con el listado completo en impresora —la cinta si es posible enviarla, mejor, la devolveremos inmediatamente con otra de regalo. Enviad todas vuestras colaboraciones a nuestra editorial en Madrid.

IMPORTANTE: Recordamos que si no especificáis que por razones personales os queréis que se publique vuestra dirección, todos los colaboradores aparecerán con nombre y dirección para que podáis ponerlos en contacto unos con otros.

Distribuidores: nota especial para vosotros

Veréis en CARTA BLANCA (página 45) que muchos lectores piden información sobre hard, soft, juegos, etc. Enviadnos listas e información **siempre con precios**. En Galería de Soft (páginas 2 y 59) pretendemos publicar reseñas de paquetes —enviados una copia de los programas que deseéis se reseñen para su comprobación. La editorial se reserva el derecho de decisión sobre la publicación de las reseñas. Siempre se publicará el nombre del distribuidor correspondiente.

Suscripciones por teléfono

Podéis suscribiros por teléfono (91) 259 54 78 preguntando por Valerie o María.

Anuncios en la revista

PROVEEDORES: anunciar vuestros productos compatibles con micros Commodore (software, periféricos, interfaces).

TIENDAS DE MICROS Y PRODUCTOS: Anunciar vuestros locales para que nuestros lectores puedan saber a dónde dirigirse.

Llamando a Barcelona: 212 73 45 y preguntar por Neus.

Llamando a Madrid: 259 54 78 y preguntar por Enrique.

Copias de nuestro predecesor Club Commodore

Nosotros no disponemos de copias originales de Club Commodore ya que Microelectrónica y Control las tiene agotadas. Por eso ofrecemos un servicio de fotocopias. Los que deseéis estas fotocopias, enviad por favor el boletín de la página 5. No hay ningún problema en realizar suscripciones por teléfono, sin embargo, el servicio de fotocopias realizarlo, por favor, por escrito. Unos quieren toda la colección, otros desean números sueltos y por escrito no puede haber malos entendidos. Darnos un margen de dos semanas para su entrega.

EJEMPLARES ATRASADOS DE «CLUB COMMODORE»

Para poder satisfacer la creciente demanda de números atrasados de nuestra Revista, agotada en todas sus ediciones, hemos puesto en marcha un Servicio para suministrar fotocopias de los ejemplares que nos sean solicitados. Para recibir las fotocopias de una o de varias ediciones, no hay más que enviarnos el boletín con los datos indicados.

SERVICIO DE FOTOCOPIAS.— NUMERO DE LA EDICION SOLICITADA.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

(Poner una X debajo del número de edición pedido)

Peticionario: D.

Calle nº

Población D.P. Provincia

Forma de pago por cheque

Precio de la edición fotocopiada: 250 ptas.

La colección completa del 0 al 15: 2.500 ptas. + 100 ptas. por gastos de envío

Incluyo cheque por ptas.

BOLETIN DE SUSCRIPCION — commodore world

NOMBRE EDAD

DIRECCION

POBLACION (.....) PROVINCIA

TELEF. MARCA Y MODELO DEL ORDENADOR

CIUDAD DONDE LO COMPRO DISTRIBUIDOR

APLICACIONES A LAS QUE PIENSA DESTINAR EL EQUIPO

Deseo iniciar la suscripción con el nº

Adjunto cheque de 2.200 pesetas

Reembolso más gastos del mismo

al recibir el primer nº de la suscripción

(Enviar a la dirección del dorso)

Extranjero \$35. Solamente por correo aéreo

Firma,

DESEO SUSCRIBIRME A **COMMODORE WORLD** POR UN AÑO AL PRECIO DE 2.200 PTS. DICHA SUSCRIPCION ME DA DERECHO, NO SOLO A RECIBIR LA REVISTA (ONCE NUMEROS ANUALES) SINO A PARTICIPAR EN LAS ACTIVIDADES QUE SE ORGANICEN EN TORNO A ELLA Y QUE PUEDEN SER COORDINACION DE CURSOS DE BASIC, INTERCAMBIOS DE PROGRAMAS, CONCURSOS, ETC.

Clave para interpretar los listados

Todos los listados que se publican en esta Revista han sido ejecutados en el modelo correspondiente de la gama de ordenadores COMMODORE. Para facilitar la edición de los mismos en la Revista y para mejorar su legibilidad por parte del usuario, se les ha sometido a ciertas modificaciones mediante un programa escrito especialmente para ello. Para los programas destinados a los ordenadores VIC-20 y COMMODORE 64, en los que se usan frecuentemente las posibilidades gráficas del teclado, se han sustituido los símbolos gráficos que aparecen normalmente en los listados por una serie de letras entre corchetes [] que indican la secuencia de teclas que se deben pulsar para obtener el carácter deseado. A continuación se da una tabla para aclarar la interpretación de las indicaciones entre corchetes:

[CRSRD] = Tecla cursor hacia abajo (sin SHIFT)

[CRSRU] = Tecla cursor hacia arriba (con SHIFT)

[CRSRR] = Tecla cursor a la derecha (sin SHIFT)

[CRSRL] = Tecla cursor a la izquierda (con SHIFT)

[HOME] = Tecla CLR/HOME (sin SHIFT)

[CLR] = Tecla CLR/HOME (con SHIFT)

Las indicaciones [BLK] a [YEL] corresponden a la pulsación de las teclas de 1 a 8 junto a la tecla CTRL. Lo mismo sucede con [RVSON] y [RVSOFF] respecto a la tecla CTRL y las teclas 9 y 10.

El resto de las indicaciones constan de la parte COMM o SHIF seguidas de una letra, número o símbolo —por ejemplo [COMM+] o [SHIFA]—. Esto indica que para obtener el gráfico necesario en el programa deben pulsarse simultáneamente las teclas COMMODORE (la que lleva el logotipo) o una de SHIFT y la tecla indicada por la letra, el número o el símbolo, en el ejemplo anterior: COMMODORE y + o SHIFT y A, respectivamente.

En los signos gráficos además se cuenta el número de veces que aparece. Por ejemplo, [7 CRSRR] equivale a 7 pulsaciones de la tecla cursor a la derecha y [3 SPC] tres pulsaciones de la barra espaciadora.

INDICE ANUNCIANTES

BASIC MICRO	21
CASA DE SOFTWARE	49
COMMODORE 64	60
COMMODORE WORLD	16
(Suscripciones)	
EAF	29
ELEKTROCOMPUTER	17
MARKETCLUB	58
MICROSISTEMAS	25,42
TECNHEL	39
VIC-20	33

Commodore
WORLD

Pedro Muguruza, 4-8º B
Teléf.: 259 54 78
MADRID-16

Sant Gervasi de Cassoles, 39-despacho 4
Teléf.: 212 73 45
BARCELONA-22

Commodore
WORLD

Pedro Muguruza, 4-8º B
Teléf.: 259 54 78
MADRID-16

Sant Gervasi de Cassoles, 39-despacho 4
Teléf.: 212 73 45
BARCELONA-22



“Mathquiz”

Diviértete con las matemáticas

Esto es un programa sencillo que utiliza las características del VIC-20 ó el Commodore 64 (gráficos, colores, cronometraje y sonido) para servir de repaso de las matemáticas estudiadas en la escuela.

COMMODORE 64

o

VIC-20

(sin ampliación)

La rutina en “Mathquiz”, un programa escrito para ejecutarse en el VIC-20 no ampliado, es una forma modernizada de las antiguas tarjetas que comprobaban la habilidad del niño para sumar, restar o multiplicar. Dado que el programa presenta no sólo problemas cuyas respuestas normalmente se memorizan, sino también problemas que tienen que ser calculados, se amplían y se refuerzan las técnicas matemáticas aprendidas en los primeros años de la escuela.

Listado 1

MATHQUIZ VIC 20

```

1 REM MATHQUIZ PARA EL VIC
2 CL=6:REM COLOR DE LOS NUMEROS
3 DEF FNL(X)=X+228R+C-PP
10 S2=36875:V=52+3:54=52+2
30 GOTO900
99 REM CANCION
100 POKEV,15
102 IFCA=6=>THEN210:REM LA CANTIDAD DE CANCION TOCADA DEPENDE DE LA PUNTUACION
103 REM SI LA NOTA ES -1 ENTONCES PARAR
105 READ P:IF P=-1 THEN202
120 READ D:POKE S2,P
160 FOR N=1TO150D:NEXTN:REM CAMBIAR 150 PARA ALTERAR EL RITMO
170 POKE S2,0:FORN=1TO20:NEXTN
190 GOTO105
202 CA=CA-1:GOTO102
210 POKE V,0:RESTORE:GOTO1600
299 REM TONO,RITMO
300 DATA103,2,195,1,215,3,209,2,195,1,107,3
310 DATA103,2,103,1,103,1,107,1,195,1,201,3,195,3,-1
320 DATA103,2,195,1,215,3,209,2,195,1
340 DATA107,3,103,2,195,1,195,1,201,1,207,1,209,3,209,3,-1
350 DATA215,1,5,195,0,5,195,1,207,1,201,1,195,1,103,2
360 DATA195,1,209,3,201,2,209,1,215,2,209,1
370 DATA207,3,195,3,-1,103,2,195,1,215,3,209,2,195,1
380 DATA107,3,103,2,195,1,195,1,201,1,207,1,209,3,209,3,-1
900 PRINT"CLRL":PRINT"SPCISUMA=S":PRINT"SPCRESTA=R"
905 PRINT"SPCIMULTIPLICACION=M":PRINT"SPCELECCION(SPC)?"
910 GETL:IFL=L*""THENV10
920 IFL="R"THEN CH=1:GOTO1000
930 IFL="S"THEN CH=1:GOTO1000
940 IFL="M"THEN CH=0:GOTO1000
950 GOTO900
1000 PRINT"CLRL":PRINT"NIVEL(SPC)DE(SPC)DIFICULTAD?":PRINT"1,2,0(SPC)3":PRINT"
(1(SPC)IES(SPC)MAS(SPC)FACIL)"
1005 PRINT"ELECCION(SPC)?"
1010 GETL:IFL=L*""THEN1010
1020 IFL="1"ORL="2"ORL="3"THEN1010
1030 F=10*(VAL(L)-1)
1060 CA=0:FORI=1TO10:PRINT"CLRL"
1070 K=INT(RND(1)*F*10):REM ELEGIR EL PROBLEMA
1074 F1=F
1075 IFCH=0ANDF1 THENF1=F/10
1080 L=INT(RND(1)*F1*10)
1090 ON CH+1GOTO1110,1120
1100 SN=45:IFL>KTHEN1070:REM ELIMINA RESTAS CON RESULTADO NEGATIVO
1105 ANS=K-L:GOTO1130
1110 SN=24:IF INT(L/10)=0 OR INT(L/10)=L/10 THEN 1110
1112 GOTO1000
1110 ANS=K*L:IF INT(ANS/1000)>0 THEN1000
1119 GOTO1130
1120 SN=43:ANS=K*L
1130 K$=STR$(K):L$=STR$(L):L$=" "4SHIFTD":L$=" "46PC":REM REPRESENTAR PROBLEMA EN PANTALLA
1140 R=6:C=11:Z$=K$:GOSUB3000
1150 R=R+1:Z$=L$:GOSUB3000
1160 C=C-LEN(L$)-1:PP=0:POKEFNL(7600),SN:POKEFNL(38400),CL
1170 R=R+1:C=11:Z$=L$:GOSUB3000
1180 R=R+1:Z$=L$:GOSUB3000
1190 MM=7600+228R+C
1200 Z1=INT(TI/100)
1210 GOSUB2210
1220 IFZIP=1THEN2000:RENTIEMPO AGOTADO
1230 POKEMM,ASC(AZ$):MM=MM-1:AP=VAL(AZ$)
1250 IFLEN(STR$(ANS))<3THEN1440
1260 GOSUB2210:IFZIP=1THEN2000
1265 IFZAP=1THEN1210
1270 POKEMM,ASC(AZ$):MM=MM-1:AP=AP+100*VAL(AZ$)
1280 IFLEN(STR$(ANS))<4THEN1440
1290 GOSUB2210:IFZIP=1THEN2000
1295 IFZAP=1THEN1260
1300 POKEMM,ASC(AZ$):MM=MM-1:AP=AP+1000*VAL(AZ$)
1310 IFLEN(STR$(ANS))<5THEN1440
1320 GOSUB2210:IFZIP=1THEN2000
1325 IFZAP=1THEN1290
1330 POKEMM,ASC(AZ$):AP=AP+10000*VAL(AZ$)
1440 IFAP=ANSTHENPRINT"25PCBIEN-MAS(SPC)ACERTADO":CA=CA+1:GOTO1490
1450 PRINT"25PCIND-LA(SPC)RESPUESTA(SPC)ES":PRINTANS
1490 FORDL=1TO750:GETL$NEXTDL,1
1500 PRINT"CLRL":PRINT"SPCHAS(SPC)ACERTADO":PRINT"SPC1"CAI"SPCDE(SPC)DIEZ (SPC)VECES"
1505 IFCA=4THENCA=4
1510 ONCA-4GOTO1530,1540,1550,1560,1570,1580
1520 PRINT"PRINT"ASEGURATE(SPC)QUE(SPC)COLOQUES"1"25PCIEL(SPC)NUMERO(SPC)DONDE(SPC)SE"
1525 PRINT"SPC"ENCUENTRA(SPC)EL(SPC)SIMBOLO(SPC)DE(SPC)DELLANTE":GOTO1600
1530 PRINT"PRINT"SPCOK--":PRINT"SPCINTENTA(SPC)UNOS(SPC)MAS!!":GOTO1600
1540 PRINT"PRINT"SPCBIEN--":PRINT"25PCINTENTA(SPC)UNOS(SPC)MAS!!":GOTO1600
1550 PRINT"PRINT"SPCMUY(SPC)BIEN--":PRINT"SPCINTENTA(SPC)UNOS(SPC)MAS!!":GOTO1600
1560 PRINT"PRINT"SPCExcelente--":PRINT"SPCINTENTA(SPC)UNOS(SPC)MAS!!":GOTO1600
1570 PRINT"PRINT"SPCFABULOSO--":PRINT"SPCINTENTA(SPC)UNOS(SPC)MAS!!":GOTO1600
1580 PRINT"PRINT"SPCFANTASTICO!--":PRINT"SPCINTENTA(SPC)UNOS(SPC)MAS!!":GOTO1600
1600 PRINT"PRINT"QUIERES(SPC)MAS?(SPC)SI(SPC)NO(SPC)?"
1610 GETD:IFD$="S"THEN1610
1620 IFD$="S"THEN900
1630 END
2000 PRINT"SPC"TIEMPO(SPC)AGOTADO":PRINT"LA(SPC)RESPUESTA(SPC)ERA"ANS
2010 POKE$4,250:POKEV,15:FORJJ=1TO50:NEXTJJ:POKE$4,0:POKEV,0:GOTO1490
2210 ZIP=0:ZAP=0:GETAZ$:IFINT(TI/100)>Z1+5*F THEN2250
2215 KK=KK+1:IFKK/2=INT(KK/2) THENPOKEMM,70:GOTO2220
2217 POKEMM,102
2220 IFAZ$="" THEN2210
2222 IFASC(AZ$)=136 THENCL=CL+1:REM CAMBIA EL COLOR
2223 IFCL<20RCL>0 THENCL=2
2224 REM BORRAR
2225 IFASC(AZ$)=29 THENMM=MM+1:AP=AP-10*(7809-MM)*(PEEK(MM)-40):ZAP=1:RETURN
2230 IFASC(AZ$)<40 OR ASC(AZ$)>57 THEN2210
2240 RETURN
2250 ZIP=1:RETURN
3000 FORPP=0TOLEN(Z$)-1:POKEFNL(7600),ASC(MID$(Z$,LEN(Z$)-PP,1))
3010 POKEFNL(38400),CL:NEXTPP:RETURN

```

Existen tres niveles de dificultad para cada operación matemática (1, 2 ó 3), siendo el nivel 1 el más fácil. Estos niveles corresponden a la suma o resta de números de uno, dos o tres dígitos. Para la multiplicación, los niveles corresponden al producto de dos números de un solo dígito, un número de dos dígitos por un número de un solo dígito, o un número de tres dígitos por un número de dos dígitos divisible por diez.

El programa selecciona cada uno de los dos números según el nivel de dificultad elegido, pero no permite que se dé un número negativo en una operación de resta. Si la respuesta dada por el estudiante no es correcta, el programa presenta una nota y la solución correcta. El estudiante dispone de un tiempo fijo para solucionar cada problema, y se permite más tiempo para los problemas más difíciles. Si el tiempo se agota, un sonido

**Nunca ha sido
aprender como**

(beep) le indica al estudiante que mire la pantalla, donde se representa la solución correcta.

El sonido de la música

Cuando se han estudiado diez problemas, la puntuación del estudiante aparece en pantalla, junto con un premio musical. La canción que aparece en el programa es la preferida de mi hija, Kathy, pero suele ser sustituida por cualquier canción. Esta parte del programa (las líneas 100-210) ha sido sacada del manual del VIC-20. La canción (las líneas 300-380) se divide en cuatro partes (separadas mediante un -1), y se toca toda la canción cuando se da una puntuación perfecta. Para cada pregunta no acertada, la canción dura menos, proporcionando así un incentivo para que todas las respuestas sean correctas.

Cuando se ejecuta, el programa pide una operación matemática (suma, resta o multiplicación), seguida por el

nivel de dificultad (1, 2 ó 3). A continuación presenta el problema en pantalla en posición vertical y sitúa el cursor debajo del dígito de la extrema derecha para que se saque el resultado de dicha columna. Entonces el cursor se va desplazando a la izquierda hasta que se introduzcan todos los dígitos.

Por lo tanto, se daría la respuesta de la suma nueve más cinco, tecleando un cuatro seguido por un uno. El resultado se introduce de derecha a izquierda para simular el proceso utilizado cuando se escribe el problema sobre papel, y es imprescindible hacerlo así cuando se estudian problemas con muchos dígitos cuyas soluciones no han sido memorizadas. Se proporciona una "goma de borrar" para que se pueda cambiar un número mediante el movimiento del cursor. En el ejemplo presentado antes, si se hubiera tecleado un uno primero, se podría borrar mediante la pulsación

tan divertido n el "mathquiz"

de la tecla apropiada del cursor, permitiendo la introducción de un nuevo número.

Por supuesto, el tiempo que se tarda en realizar tales correcciones se descuenta del tiempo asignado para cada problema. Si el estudiante empieza a perder el interés, se puede modificar el color del siguiente problema mediante la pulsación de la tecla f7. La canción también puede ser cambiada, preferentemente por el estudiante.

El Listado 1 presenta el programa para el VIC-20, y el Listado 2 para el Commodore 4. Los niños disfrutarán "jugando con el ordenador" mientras refuerzan sus conocimientos matemáticos con este programa.

Listado 2

```

1 REM MATHQUIZ PARA EL C-64
2 CL=1:REM COLOR DE LOS NUMEROS
3 DEF FNL(X)=X+48+C-PP
10 S1=54277:S2=54276:S3=54273:S4=54272:V=54296
30 GOTO900
99 REM CANCION
100 POKEV,15:POKE$1,9
102 IFCA=6<0THEN210:REM LA CANTIDAD DE CANCION TOCADA DEPENDE DE LA PUNTUACION
103 REM $1 LA NOTA ES -1 ENTONCES PARAR
105 READ P:IF P=-1 THEN202
120 READPP,D:POKE$3,P:POKE$4,PP:POKE$2,17
160 FOR N=1TO1000D:NEXTN:REM CAMBIAR 1000 PARA ALTERAR EL RITHO
170 POKE$2,16:FORN=1TO20:NEXTN
190 GOTO105
202 CA=CA-1:GOTO102
210 POKE V,0:RESTORE:GOTO1600
299 REM TONO,RITHO
300 DATA14,24,2,16,195,1,25,30,3,22,96,2,16,195,1,14,239,3
310 DATA14,24,2,14,24,1,14,24,1,14,239,1,16,195,1,18,209,3,16,195,3,-1
320 DATA14,24,2,16,195,1,25,30,3,22,96,2,16,195,1,14,239,3
340 DATA14,24,2,16,195,1,16,195,1,18,209,1,21,31,1,22,96,3,22,96,3,-1
350 DATA25,30,1,5,16,195,1,25,30,3,22,96,1,21,31,1,18,209,1,16,195,1,14,24,2
360 DATA16,195,1,22,96,3,18,209,2,22,96,1,25,30,2,22,96,1,21,31,3,16,195,3,-1
370 DATA14,24,2,16,195,1,25,30,3,22,96,2,16,195,1,14,239,3
380 DATA14,24,2,16,195,1,16,195,1,18,209,1,21,31,1,22,96,3,22,96,3,-1
900 PRINT*CLR*:PRINT*{SPC}SUMA=S*:PRINT*{SPC}RESTA=R*
905 PRINT*{SPC}MULTIPLICACION=M*:PRINT*{SPC}ELECCION{SPC}7*
910 GETL$:IFL$=""THEN910
920 IFL$="R"THEN CH=-1:GOTO1000
930 IFL$="S"THEN CH=1:GOTO1000
940 IFL$="M"THEN CH=0:GOTO1000
950 GOTO900
1000 PRINT*CLR*:PRINT*NIVEL{SPC}DE{SPC}DIFICULTAD?:PRINT*1,2,0{SPC}3*:PRINT*
1100 IF{SPC}IES{SPC}MAS{SPC}I{SPC}I*
1005 PRINT*ELECCION{SPC}7*
1010 GETL$:IFL$=""THEN1010
1020 IFL$="1"ORL$="2"ORL$="3"THEN1010
1030 F=10*(VAL(L$)-1)
1040 CA=0:FORI=1TO10:PRINT*CLR*I*
1070 K=INT(RND(1)*F*10):REM ELEGIR EL PROBLEMA
1074 FI=F
1075 IFCH=0ANDF<1THENFI=F/10
1080 L=INT(RND(1)*F*10)
1090 ON CH+1GOTO1110,1120
1100 SN=45:IFL$="S"THEN1070:REM ELIMINA RESTAS CON RESULTADO NEGATIVO
1105 ANS=K-L:GOTO1130
1110 SN=24:IF INT(L/10)=0 OR INT(L/10)=L/10 THEN 1110
1112 GOTO1080
1116 ANS=K+L:IFINT(ANS/10000)>0THEN1080
1119 GOTO1130
1120 SN=43:ANS=K+L
1130 K$=STR$(K):L$=STR$(L):L$=" "[45SHIFT0]*L$+"{4SPC}":REM REPRESENTAR PROBLEMA EN PANTALLA
1140 R=6:C=11:Z$=K:GOSUB3000
1150 R=R+1:Z$=L:GOSUB3000
1160 C=C-LEN(L$)-1:PP=0:POKEFNL(1024),SN:POKEFNL(55296),CL
1170 R=R+1:C=11:Z$=L:GOSUB3000
1180 R=R+1:Z$=L:GOSUB3000
1190 MM=1024+40*R+C
1200 Z1=INT(TI/100)
1210 GOSUB2210
1220 IFZ1P=1THEN2000:REMITIEMPO AGOTADO
1230 POKEHH,ASC(AZ$):MM=MM-1
1240 AP=VAL(AZ$)
1250 IFLEN(STR$(ANS))<3THEN1440
1260 GOSUB2210:IFZ1P=1THEN2000
1265 IFZ1P=1THEN1210
1270 POKEHH,ASC(AZ$):MM=MM-1:AP=AP+10*VAL(AZ$)
1280 IFLEN(STR$(ANS))<4THEN1440
1290 GOSUB2210:IFZ1P=1THEN2000
1295 IFZ1P=1THEN1260
1300 POKEHH,ASC(AZ$):MM=MM-1:AP=AP+100*VAL(AZ$)
1310 IFLEN(STR$(ANS))<5THEN1440
1320 GOSUB2210:IFZ1P=1THEN2000
1325 IFZ1P=1THEN1290
1330 POKEHH,ASC(AZ$)
1335 AP=AP+1000*VAL(AZ$)
1440 IFAP=ANSTHENPRINT*{2SPC}BIEN-HAS{SPC}ACERTADO*:CA=CA+1:GOTO1490
1450 PRINT*{2SPC}NO-LA{SPC}RESPUESTA{SPC}IES*:ANS
1490 FORDL=1TO750:GETL$:NEXTDL,II
1500 PRINT*CLR*:PRINT*{SPC}HAS{SPC}ACERTADO*:PRINT*{SPC}*{CA}*{SPC}DE{SPC}DIEZ {SPC}VECES*
1505 IFCA<4THENCA=4
1510 ONCA-4GOTO1530,1540,1550,1560,1570,1580
1520 PRINT
1521 PRINT:PRINT*{SPC}ASEGURATE{SPC}QUE{SPC}COLOQUES{SPC}EL{SPC}NUMERO{SPC}DONDE{SPC}*
1525 PRINT*{2SPC}ISE{SPC}JENCENTRAL{SPC}EL{SPC}SIMBOLO{SPC}DESTELLANTE*:GOTO1400
1530 PRINT:PRINT*{SPC}OK--*:PRINT*{SPC}INTENTA{SPC}JUNOS{SPC}JUNOS*:GOTO1400
1540 PRINT:PRINT*{SPC}BIEN--*:PRINT*{2SPC}INTENTA{SPC}JUNOS{SPC}JUNOS*:GOTO1400
1550 PRINT:PRINT*{SPC}BIEN--*:PRINT*{SPC}INTENTA{SPC}JUNOS{SPC}JUNOS*:GOTO1400
1560 PRINT:PRINT*{SPC}EXCELENTE--*:PRINT*{SPC}INTENTA{SPC}JUNOS{SPC}JUNOS*:GOTO1400
1570 PRINT:PRINT*{SPC}FABULOSO--*:PRINT*{SPC}INTENTA{SPC}JUNOS{SPC}JUNOS*:GOTO1400
1580 PRINT:PRINT*{SPC}FANTASTICO--*:PRINT*{SPC}INTENTA{SPC}JUNOS{SPC}JUNOS*:GOTO1400
1600 PRINT:PRINT*QUERES{SPC}MAS?{SPC}{SI{SPC}O{SPC}NO)*
1610 GETD$:IFD$=""THEN1610
1620 IFD$="S"THEN900
1630 END
2000 PRINT*{SPC}TIEMPO{SPC}AGOTADO*:PRINT*LA{SPC}RESPUESTA{SPC}IERA*:ANS
2010 FORN=15TO80STEP1:POKEV,N:POKE$2,129:POKE$1,15:POKE$3,40:POKE$4,200:NEXTN
2020 POKE$2,0:POKE$1,0:GOTO1490
2210 Z1P=0:ZAP=0:GETAZ$:IFINT(TI/100)>Z1+5*FTHEN2250
2215 KK=KK+1:IFKK/2=INT(KK/2)THENPOKEHH,70:GOTO2220
2217 POKEHH,102
2220 IFAZ$=""THEN2210
2222 IFASC(AZ$)=136THENCL=CL+1:IFCL=6THENCL=CL+1:REM CAMBIA EL COLOR
2223 IFCL<20RCL=0THENCL=1
2224 REM BORRAR
2225 IFASC(AZ$)=29THENMM=MM+1:AP=AP-10*(1395-MM)*(PEEK(MM)-40):ZAP=1:RETURN
2230 IFASC(AZ$)<40 OR ASC(AZ$)>57THEN2210
2240 RETURN
2250 Z1P=1:RETURN
3000 FORPP=0TOLEN(Z$)-1:POKEFNL(1024),ASC(MID$(Z$,LEN(Z$)-PP,1))
3010 POKEFNL(55296),CL:NEXTPP:RETURN

```

Disfruta mucho más con la magia de la conversión (y un juego de demostración)

Te sientes marginado porque tienes un Commodore 64, y dispones de poco software? En este artículo te indican la forma de convertir los numerosos programas del VIC-20 para que se procesen en el C-64.

Al principio mi interés en convertir programas fue provocado por la escasez de programas para el Commodore 64 y la facilidad con que se podían conseguir para el VIC-20. Un motivo secundario fue que me proporcionaba la oportunidad para comprender mejor el lenguaje Basic y para estudiar las técnicas de programación que se utilizan en la animación. Cualquiera que sea el motivo, primero hay que elegir un programa para convertir.

Naturalmente, la mejor forma de elegir un programa es verlo funcionar en un VIC-20. Aunque esto no sea siempre posible, una vez elegido el programa, éste tiene que introducirse en el C-64 exactamente igual que aparece en el listado, y se tiene que salvar antes de ser procesado.

Este es el programa inicial. Si se puede comprobar en un VIC, mucho

mejor y si se detecta algún fallo resulta útil apuntar el número de la sentencia. Es probable que los mismos fallos salgan en el C-64.

Las Principales Diferencias de Programa

El VIC-20 y el C-64 se parecen en muchos aspectos, pero existen diferencias en los métodos que emplean para controlar los registros utilizados para producir la acción en la pantalla y en el sonido. Los procedimientos de lectura del teclado y del joystick también son diferentes. (Las Tablas 1 y 2 presentan un resumen).

Las sentencias "Print" o "Poke" controlan los caracteres de la pantalla y los colores de los caracteres. Las sentencias "Print" en un VIC se pueden imprimir en la pantalla de un C-64, realizando unas modificaciones mínimas dada la diferencia del tamaño de la pantalla. Normalmente, un cambio es necesario si una sentencia "Print" termina en punto y coma, ya que se escribe de forma diferente en la pantalla más ancha del C-64.

Las sentencias "Poke" siempre se tienen que modificar para que se ajuste la posición de memoria del VIC a la posición equivalente en el C-64. Yo he inventado un programa sencillo (El Listado 1) que convierte la posi-

Si tiene un C-64
pero no suficientes programas
puedes coger algunos
del VIC
y diviértete convirtiéndolos

Listado 1.

```
1000 PRINT"CLR":REM***CAMBIO DE DIRECCIONES DE PANTALLA DEL VIC 20 AL C-64
1010 POKE53201,1:FLAG=0:VA=0:INPUT"*(SPC)DIRECCION(SPC)VIC(SPC)20*(VA:IFVA=0THE
NSTOP
1011 IFVA<7680THENPRINT"(RVSON)DIRECCION(SPC)DEMASIADO(SPC)BAJA":GOTO1010
1012 IFVA>8195ANDVA<38400THENPRINT"(RVSON)DIRECCION(SPC)NO(SPC)VALIDA":GOTO1010
1013 IFVA>38905THENPRINT"(RVSON)DIRECCION(SPC)DEMASIADO(SPC)ALTA":GOTO1010
1015 IFVA>38399ANDVA<38906THENVA=VA-38720:FLAG=1
1020 RO=INT((VA-7680)/22)
1030 CO=VA-7680-22*RO
1040 CA=1024+RO*40+CO:IFFLAG=1THENCA=CA+54272
1050 PRINTTAB(21)CHR$(145)"*DIRECCION(SPC)C-64*(CA:GOTO1010
```


ción de caracteres en el VIC en la posición para el C-64. Se puede determinar la posición del carácter "Poke" en color añadiendo un "offset" a la posición del carácter. El "offset" para el VIC es 30720 y el "offset" para el C-64 es 54272.

En el VIC un sólo registro controla los colores del borde y el fondo de la pantalla, mientras en el C-64 se emplean dos registros para controlar los colores. El formato de la sucesión de los colores de los "Poke" utilizado por el VIC es el más complicado. El valor más bajo de "Poke" es 8 y representa un fondo negro con un borde negro.

A medida que este valor se incrementa por uno, el color del borde se incrementa por uno en el orden de la clave de color en el teclado (de negro a amarillo). Para aumentar el color del fondo por uno (en el orden presentado en la Tabla 1), se necesita un incremento de 16. Por ejemplo, 24 produce un borde negro con un fondo blanco; el "Poke" más alto, 255, produce un fondo de amarillo claro y un borde amarillo.

Dado que los códigos de color "Poke" en el C-64 son constantes (0-15), el efecto producido se ve en seguida que se hace un "Poke" cada número de color en los dos registros. Cuando se hace un "Poke" cero en el registro del borde en el C-64, aparece un borde negro.

El VIC-20 y el C-64 también producen el sonido de dos maneras diferentes. El VIC tiene cuatro registros dedicados al sonido. Tres de las voces son musicales, cada una con un determinado alcance de frecuencias, mientras que la cuarta produce ruido al azar. Las voces del VIC pueden estar en "on" o en "off", y se controlan especialmente por las posiciones del volumen y de la frecuencia (128-255).

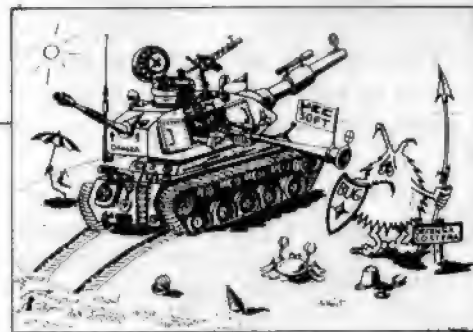
El C-64 posee un sintetizador de sonido de tres voces. Cada voz es capaz de producir ruidos y música. El sonido exacto que se produce se controla mediante las posiciones de volumen, frecuencia, forma de ondas, principio/final y mantener la tecla pulsada/soltar la tecla. El grado de control alcanzable sobrepasa el que se necesita para la mayoría de los juegos o programas.

Diferencias de Entrada

Físicamente los teclados de los dos ordenadores son idénticos, pero al pulsar una tecla determinada, los

registros que almacenan la información (posiciones 197-203) contienen distintos valores. Dado que la mayoría de los programas leen el teclado, se necesitan unos cambios de programa para producir el efecto deseado. Es muy fácil cambiar la sentencia "IF" que se utiliza para detectar la pulsación de una tecla determinada. Los valores para las teclas utilizados normalmente en el VIC y el C-64 se presentan en la Tabla 2.

Se aprecian mejor las diferencias en las técnicas de lectura para el mando (joystick) en las dos máquinas si se



consideran unos ejemplos típicos de la aplicación de programas. Uno de estos ejemplos es "Dr. Dementia", un programa que yo convertí y que se publicó en "Microcomputing" en agosto de 1983.

Este programa utiliza el mando (joystick) principalmente como un "paddle", desplazándose solamente a

Disposición de caracteres de pantalla:

VIC 22 columnas × 23 filas

C-24 40 columnas × 25 filas

Fondo de pantalla y controles:

VIC Dieciséis (negro, blanco, rojo, cyan, morado, verde, azul, amarillo, naranja, naranja claro, rosa, cyan claro, morado claro, verde claro, azul claro, amarillo claro)
Control 36879 (8-255)

C-24 Dieciséis (negro a naranja, marrón, rojo claro, gris 1, gris 2, verde claro, azul claro, gris 3)
Control 53281 (0-15)

Color de caracteres y borde y controles:

VIC Ocho (negro a amarillo)
Control de borde 36879 (8-255)

C-64 Dieciséis (todos colores de borde y fondo)
Control de borde 53280 (0-15)

Mapas de caracteres de pantalla y memoria de colores (empezando en la posición de pantalla en la parte superior a la izquierda):

VIC Carácter 7680-8185
Color 38400-38905
Carácter "offset" a color 30720

C-64 Carácter 1024-2023
Color 55296-56295
Carácter "offset" a color 54272

Chip generador de sonido y controles:

VIC Tres voces musicales (Alcance total de cinco octavos: inferior 36874, mediano 36875, superior 36876)
Una voz de ruido al azar 36877
Control de volumen 36878

C-64 Tres voces (Alcance total de ocho octavos)
Controles: volumen, frecuencia, forma de ondas, principio/final, mantener pulsada/soltar la tecla.

Disposición de teclado (Vea la Tabla 2 para comparar)

Tabla 1. Principales diferencias entre los ordenadores VIC-20 y Commodore 64.

DISFRUTA MUCHO MAS con la magia de la conversión

la izquierda y a la derecha. Se puede utilizar una programación semejante para obtener movimiento en cualquiera de las ocho direcciones. Las sentencias de programa original del VIC y el programa convertido del C-64 se presentan a continuación:

```
VIC  POKE 37154,127: PK=PEEK(37152)
      AND 128: JO=NOT(PK=0): POKE
      37154,255: PK=PEEK(37151): J2=
      NOT((PK AND 16)=0): FB=NOT((PK
      AND 32)=0)
C-64  JV=PEEK(56320) AND 15: FB=
      NOT((PEEK(56320) AND 16)=0)
```

El primer "Poke" del VIC fija uno de los dos registros de la dirección de datos en la modalidad de entrada. Esto permite la lectura del interruptor 3 (derecha) del mando (joystick) y se analiza una función lógica "AND" para establecer un valor para PX. Si el mando (joystick) se desplaza a la derecha, J0 se fijaría en uno mediante la sentencia lógica AND. Si el joystick estuviera estacionario, J0 sería cero.

Dado que el registro de la dirección de datos tiene dos funciones en el VIC, se tiene que restaurar o hacerle un POKE a 255 después de completar la lectura del interruptor 3. Los otros tres interruptores del mando (joystick) se leen haciendo un PEEK en la posición 37151. Se determina el estado de los interruptores realizando la función lógica "AND" en el valor leído por las potencias de 2. Si se realiza la función lógica "AND" en PEEK con 4 se detecta el interruptor que se desplaza hacia arriba. De modo semejante, si se realiza la función lógica AND con el 8 y 16, se detectan las direcciones hacia abajo y a la izquierda, respectivamente.

El movimiento en diagonal se detecta cuando dos interruptores se cierran a la vez. Se detecta el movimiento del botón de disparo mediante la realización de la función lógica AND con 128. Las sentencias lógicas

para J2 y FB son ejemplos de la detección de la dirección a la izquierda y el botón de disparo. El VIC dispone de otros registros para leer el mando, pero los que se presentan aquí son los que se utilizan normalmente. Los registros alternativos son 37137 y 37139.

El joystick del C-64 se lee simplemente haciendo un "PEEK" en una de las dos posiciones (56320 para la Puerta 2; 56321 para la Puerta 1). El valor donde se ha hecho el PEEK da 14, 13, 11 y 7 para el movimiento del joystick hacia arriba, hacia abajo a la derecha y a la izquierda, respectivamente cuando se realiza la función lógica "AND" con 15. El movimiento del mando (joystick) en diagonal produce valores de 5, 6, 9 y 10. El botón de disparo se detecta realizando la función lógica "AND" del PEEK con 16.

El procedimiento general para modificar una porción del programa del mando en el VIC para que funcione en el C-64 es la que se presenta a continuación:

—Explorar el listado para los registros mencionados y el valor que utilizan para "AND".

—Fijar la lógica del programa que utiliza la entrada del mando (joystick).

—Modificar las sentencias VIC a las simples sentencias equivalentes en el C-64.

Una demostración de Prueba

La mejor forma de saber la manera en que todas estas diferencias afectan un programa típico es realizando un ejercicio de conversión. Para este ejercicio, he elegido un juego de "Public Domain", "Artillery" (Artillería), escogido porque las modificaciones de conversión son típicas de la mayoría de los programas sencillos de Basic. Los Listados 2 y 3 presentan el programa original de VIC y el programa convertido del C-64. El derecho para utilizar este juego fue otorgado por Bill Munch de Public Domain, Inc., West Milton, Ohio.

El método utilizado proporciona un programa del C-64 que funciona de modo semejante que el programa del VIC. Sin embargo, el uso del procedimiento detallado no produce en el C-64 SID (Sound Interface Device) (Dispositivo de Sonido de la Interface) exactamente los mismos sonidos que los que se producen en el VIC. Si se desea mayor fidelidad, los registros SID tienen que ser ajustados, que a su vez significa ajustar los controles de

TECLAS STANDARD

FILA 1			FILA 2			FILA 3			FILA 4		
Tecla	VIC	C-64	Tecla	VIC	C-64	Tecla	VIC	C-64	Tecla	VIC	C-64
—	8	57	Q	48	62	A	17	10	Z	33	12
1	0	56	W	9	9	S	41	13	X	26	23
2	56	59	E	48	14	D	18	18	C	34	20
3	1	8	R	10	17	F	42	21	V	27	31
4	57	11	T	50	22	G	19	26	B	35	28
5	2	16	Y	11	25	H	43	29	N	28	39
6	58	19	U	51	30	J	20	34	M	36	36
7	3	24	I	12	33	K	44	37	.	29	47
8	59	27	O	52	38	L	21	42	.	37	44
9	4	22	P	13	41	:	45	45	/	30	55
0	30	35		53	46	;	22	50			
+	5	40	*	14	49	=	46	53			
—	61	43	^		54	54					
£	6	48									

Teclas de Función y Control Especial

Tecla	VIC	C-64	Tecla	VIC	C-64
Clear—Home	62	51	f1	39	4
Insert—Delete	62	0	f3	47	5
Return	15	1	f5	55	6
Space Bar	32	6	f7	63	3
Cursor—Vertical	31	7	Cursor—Horizontal	23	2

Nota: Cuando no hay teclas pulsadas las posiciones de memoria 197 y 203 contienen 64.

Tabla 2. Diferencias del teclado del VIC-20 y el C-64.

frecuencia, forma de ondas, principio/final y mantener pulsada/soltar tecla.

Lo difícil de la conversión es seguir el flujo del programa. Como los programadores del VIC disponen de una memoria de máquina muy limitada utilizan pocas sentencias "Remark". Una manera de rastreo es colocando un "Stop" antes y después de cada sentencia de bifurcación (If, Go Sub, Go To, etc.) en el programa inicial. Luego se modifican las sentencias antes del "Stop" y se retiran a medida que cada división pasa sin problemas obteniendo el proceso del programa.

En "Artillería" dos jugadores disparan cañones por encima de una montaña que cambia de forma al azar. Para que el juego resulte más complicado hay que tener en cuenta el factor del viento al elegir el ángulo de disparo del cañón. También hay que ajustar la carga, o la pólvora.

Las líneas o sentencias del listado del C-64 que se modifican serán tratadas por grupo a medida que surjan cambios comunes de sonido, pantalla, teclado, etc.

Líneas 0 y 75: La función DEF en la línea 0 se utiliza a lo largo del programa para hacer un "Poke" de varios símbolos gráficos en las posiciones de la memoria de pantalla, según los valores actuales de X e Y. Esta función DEF también podría haber sido utilizada por el programador del VIC en la línea 75, pero no se hizo así. Las diferencias de la ecuación en ambas líneas se relacionan al comienzo de la memoria de caracteres de la pantalla (VIC, 7680; C-64, 1024) y al número de filas y columnas (VIC, 22 columnas \times 23 filas; C-64, 40 columnas \times 25 filas).

Línea 4: Esta subrutina inicial (líneas 3500—3640) trata, en principio, con los registros de teclas y los "array". La primera línea establece los límites de memoria de los registros SID e inicializa tres variables (BO, BA, CU) para controlar los colores de pantalla para el borde, el fondo y el cursor. La siguiente línea borra el chip generador de sonido, SID, según las recomendaciones de Commodore.

El programa inicia el establecimiento de variables que se utilizan para seguir la pista de los registros de teclas SID; a continuación inicializa las dos voces SID utilizadas en el programa. Las voces 1 y 2 imitan las voces de música y ruidos del VIC. La voz 3 se inicializa pero no se utiliza en el programa.

Para la mayoría de los programas se pueden utilizar valores fijos para principio/final (AD), mantener pulsada/soltar la tecla (SR), y de baja frecuencia (LF). Yo utilizo el comando "Poke" estandar, y lo modifiqué solamente si la calidad del sonido es pobre. Finalmente, la subrutina fija el ángulo inicial y los valores de pólvora para cada jugador.

Línea 5: Los comandos "Poke" del C-64 desempeñan las mismas funciones que los "Poke" del VIC; fijan los colores deseados para el borde y el fondo de la pantalla y para el cursor. Como hemos mencionado antes, en el VIC el valor almacenado en la posición 36879 controla los colores del borde y el fondo.

En el VIC, los comandos "Poke" producen los colores blanco, blanco y azul para el borde, fondo y cursor. Los colores del C-64 son blanco, blanco y gris 1.

En la versión del C-64, KX se fija en 65272. Esto es el "offset" entre las posiciones de los caracteres y la memoria de la pantalla en color. Se



utilizará más adelante en el programa y se necesita para mantener la estructura del programa.

A veces, es necesario abreviar las palabras clave y añadir variables o sentencias para reducir las sentencias del VIC (longitud máxima de 88 caracteres) para que quepan en el C-64 (longitud máxima de 80 caracteres).

Línea 7: Esta subrutina mejorada (las líneas 4.000-4.100) escribe las instrucciones de juego en la pantalla. Retorna a la sentencia 7 cuando el "Peek" o el valor almacenado en la posición 197 (al pulsar una tecla) se fija en 60. Esto ocurre cuando se pulsa la barra de espaciado.

Línea 11: Se realizan mejoras en la versión del C-64 para fijar los valores actuales de ángulo (A) y pólvora (V) a los valores antes seleccionados por el jugador 1 ó 2. Los parámetros actua-

VIC Address	Función	C-64 Address
197 ó 203	Tecla pulsada	197 ó 203
646	Color del cursor	646
7680—8185	Posición de memoria de caracteres de pantalla	1024—2023
36874-36878	Registro de sonido	54272—54296
36879	Registro de color de pantalla (fondo y borde)	53280 y 53281
37137	Registro de salida alternativa del mando (joystick)	—
37139	Registro de dirección de datos alternativa del mando (joystick)	—
37151	Registro de salida principal del mando (joystick) para interruptores 0, 1 y 2	56320 y 56321 (all switches)
37152	Registro de salida principal del mando (joystick) para interruptor 3 (derecha)	—
37154	Registro de dirección de datos principal del mando (joystick)	—
38400—38905	Posiciones de memoria de color de caracteres de pantalla	55296—56295

Tabla 3. Direcciones importantes del VIC-20 y las equivalencias en el C-64.

DISFRUTA MUCHO MAS con la magia de la conversión

les se salvan en la sentencia 27 y se inicializan en la sentencia 3630.

Línea 13: Se necesita el "Poke" adicional en el programa del C-64 para que el cursor vuelva al color de gris 1 después de que las montañas y los cañones se imprimen en la subrutina 3200.

Líneas 14, 15, 18, 19 y 24: Estas líneas se tienen que modificar dadas las diferencias en los teclados. Un "Peek" de la posición 197 ó 203 retorna valores diferentes en los dos ordenadores cuando se pulsa la misma tecla. La Tabla 2 compara estos valores.

Línea 24: En la versión del C-64, se elimina el "Poke" de sonido equivalente al del VIC. El Poke del VIC sube el volumen al máximo, 15. Si este "Poke" no se eliminara, el SID emitiría un ruido seco y repetitivo. Se retrasa la conversión del ángulo de disparo elegido en radianes hasta que se salva como AS(PL) en la sentencia 27.

Líneas 25, 26, 65, 67, 81, 120, 2001, 2003, 2060, 2070, 2076 y 2078: Los "Poke" de sonido del VIC en estas sentencias se modificaron para reproducir la música o sonidos equivalentes. Para que el procedimiento de conversión resultara fácil, decidí fijar la mayoría de los registros SID en la subrutina 3500; sin embargo, se tienen que controlar un mínimo de tres. Esto es un registro más que los que se necesitan en el VIC.

Los registros para el C-64 son VL (volumen), HF (alta frecuencia) y WF (forma de ondas). Los registros del VIC son volumen (36878) y frecuencia, o voz, (36874-36877). Los registros de volumen de ambos ordenadores se manejan de forma idéntica, pero los registros de frecuencia tienen diferentes alcances de "Poke". Se puede hacer un "Poke" en los registros del VIC con valores de entre 128 y 255, mientras que el alcance del C-64, de 0 a 255, es el doble de grande.

Los registros de forma de ondas del C-64 se utilizan para seleccionar el tipo de sonido producido y para apagarlo y encenderlo. Yo fijé la voz 1 para ruido y la voz 2 para música. La forma de onda para la voz 1, WF(1), contiene o 129 (ruido encendido) o 128 (ruido apagado). De modo semejante, el registro de forma de ondas de la voz 2 contiene o 17 (música encendida) o 16 (música apagada).

Los registros de frecuencia de VIC controlan el tipo de sonido producido. Los tres primeros registros (el registro más bajo es la voz del octavo

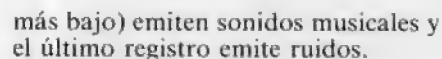
El Listado 2. Programa original "Artillería" para el VIC-20. Este listado no funciona en un VIC-20 no ampliado a no ser que se quiten las sentencias REM.

Artillería VIC 20.

```

8 CLR:DEFFNCL(R)=7680.5*INT(22.5-Y)*22+X:X=NRND(-1):PL=1
1 REM:ESTO ES ARTILLERIA 20 PROGRAMA ORIGINAL DE "PUBLIC DOMAIN" VER COMENTARIO
5 PRINT"(CLR)":POKE36879,25
9 GOSUB3200:W=(2*NRND(1)):W=W/2:R=INT(2*NRND(1)):IFR=1THENW=-W
10 G=V.0:H=.04:T=0:VX=0:VY=0:X=0:Y=0
11 R=INT(2*NRND(1)):LP=8:A=45:V=28:IFPL=2THENLP=11
12 RR=(NRND(1)):R=INT(2*NRND(1)):IFRR>.2THENRR=-.2
13 GOSUB3200
14 IFPEEK(197)=39THENA=A+1
15 IFPEEK(197)=47THENA=A-1
16 IFA<0THENA=0
17 IFA>90THENA=90
18 IFPEEK(197)=55THENV=V+1
19 IFPEEK(197)=63THENV=V-1
20 IFV<1THENV=1
21 IFV>40THENV=40
22 PRINT"(HOM)"SPC(LP*8)"(26PC)":PRINT"(HOM)"SPC(LP)"ANGULO(SPC)=3SPC(13)CRSL1"
23 PRINT"(HOM)"CRSRD)"SPC(LP*8)"(26PC)":PRINT"(HOM)"CRSRD)"SPC(LP)"POLVORA=(3SPC(13)CRSL1)"V
24 IFPEEK(197)=15THENA=W/100+A:POKE36878,15:GOTO37
25 POKE36878,W:POKE36877,W:GOTO14
37 POKE36877,0:PRINT"(HOM)"(44SPC)"
38 PRINT"(HOM)"(2CRSRD)"(226PC)"
39 ONPLGOTO40,50
40 X=X1:Y=Y1+1:X1=X:Y1=Y:PC=FNCL(0):S=1:GOTO40
41 S=-1:X=X2:Y=Y2+1:X1=X:Y1=Y:PC=FNCL(0)
42 VY=SIN(A)*V:VX=S*(COS(A)*V):FORI=220TO16STEP-1:POKE36877,1:NEXT:POKE36877,0
43 T=T+H:X=X1+VX*T-W*12:Y=Y1+VY*T-G*12
45 PK=7680.5*INT(22.5-Y)*22+X:PK=INT(PK):POKEPC,32
46 IFY>22THEN15
47 IFX<0ORX>21THENPOKE36876,0:GOTO2004
48 PC=PK:IFY<0THEN2000
49 IFPEEK(PK)=32THEN1000
110 POKEPK+38720,0:POKEPK,46
115 IFY<2+137=V255ORY<2+137<0THEN130
120 POKE36876,Y*2+137
130 IFY>0THEN70
140 GOTO2000
1000 IFPEEK(PK)=233ORPEEK(PK-22)=233THENPOKEPK,42:H1=H1+1:H=1:PL=1:GOTO2000
110 IFPEEK(PK)=223ORPEEK(PK-22)=223THENPOKEPK,42:H2=H2+1:H=1:PL=2:GOTO2000
2000 KK=PEEK(PK):POKEPK,32:I=SIN(I):POKEPK,KK:POKEPK,KK:POKEPK,KK
2001 POKE36876,0:FORI=1TO50:POKE36877,128+I:NEXT
2002 POKEPK,32:I=SIN(I):POKEPK,KK:POKEPK,32:I=SIN(I):POKEPK,KK
2003 FORI=1TO50:POKE36877,170-I:NEXT:POKE36877,0
2004 IFPL=1THENPL=2:GOTO2010
2005 IFPL=2THENPL=1:GOTO2010
2010 IFH=1THENH=0:PRINT"(CLR)":GOTO9
2020 GOTO10
2060 K=30720:POKE36876,0:G=7:SB=36877:FORI=1TO10
2070 POKEPK+K,0:POKE36,128:G=SIN(I):POKEPK-22+K,G
2072 POKEPK-43+K,G:POKE36,150:POKEPK-45+K,G
2076 POKE36,108:G=SIN(I):G=SIN(I):IFG<42THENG=32:GOTO2070
2077 G=42:K=0
2078 NEXTI:POKE36,128:FORI=1TO100:NEXT:POKE36,0:GOTO2000
2080 G=42:K=0
3000 PRINT"(HOM)"(2CRSRD)"(RVSON)"(RED)SC="H1"(HOM)"(2CRSRD)"(1CRSRD)"(PUR)SC="H2
3003 R=INT(6*NRND(1))+1:FORV=0TOR-1:FORX=0TOS:POKEFNCL(0)+30720,5
3005 POKEFNCL(0),224:NEXTX:NEXTV
3007 VI=Y:X1=INT(4*NRND(1))+1:X1=X1:POKEFNCL(0)+30720,2:POKEFNCL(0),223
3010 FORV=0TOS:FORX=6TOS:POKEFNCL(0)+30720,5
3020 POKEFNCL(0),160:NEXTX:NEXTV:R=INT(4*NRND(1))+1:FORV=0TOR-1:FORX=6TOS:
3030 POKEFNCL(0)+30720,5:POKEFNCL(0),224:NEXTX:NEXTV
3035 Z=Y:X2=INT(3*NRND(1))+1:X2=X2:POKEFNCL(0)+30720,4:POKEFNCL(0),233
3040 P1=INT(3*NRND(1))+1:P1=P1+5:P2=INT(3*NRND(1))+1:P2=P2+5:P1=INT(2*NRND(1))+1
3050 IF1=1THENL1=255:GOTO3600
3055 L1=254
3060 I=INT(2*NRND(1))+1:IFI=1THENL2=97:GOTO3070
3065 L2=252
3070 X=P1:Y=A:POKEFNCL(0)+30720,5:POKEFNCL(0),L1
3071 FORX=P1+1TOP2-1:Y=A:POKEFNCL(0)+30720,5
3075 POKEFNCL(0),160:NEXTX:X=P2:POKEFNCL(0)+30720,5:POKEFNCL(0),L2
3080 FORV=7TO17:RR=INT(4*NRND(1))+1
3085 IFRR=1THENA1=P1+2:L1=254
3090 IFRR=2THENA1=P1+1:L1=254
3095 IFRR=3THENA1=P1:L1=225
3100 RR=INT(4*NRND(1))+1:IFRR=1THENA2=P2-2:L2=252
3110 IFRR=2THENA2=P2-1:L2=252
3120 IFRR=3THENA2=P2:L2=97
3130 IFA2<AORAL=A2THEN3160
3140 X=A1:POKEFNCL(0)+30720,5:POKEFNCL(0),L1:FORX=A1+1TOA2-1:POKEFNCL(0)+30720,5
3145 POKEFNCL(0),160:NEXTX:X=A2:POKEFNCL(0)+30720,5:POKEFNCL(0),L2
3150 P1=A1:P2=A2:IFTT=1THEN3160
3155 NEXTV
3160 PK=P1:PY=Y:RETURN
3200 IFR=1THENRR=-RR
3210 W=W*(RR/INT(4*NRND(1))+1)
3220 IFW<-2THENW=-2
3225 IFW>2THENW=2
3235 WD=240:WH=INT(ABS(W*0.1)):IFWH>15THENWH=15
3240 IFW>0THEN3270
3260 X=PK:Y=PY:POKEFNCL(0)+30720,6:POKEFNCL(0),103:Y=Y+1
3261 POKEFNCL(0)+30720,6:POKEFNCL(0),103
3262 POKEFNCL(0)+30721,6:POKEFNCL(0),102:GOTO3300
3270 X=PK:Y=PY:POKEFNCL(0)+30720,6:POKEFNCL(0),103:Y=Y+1
3271 POKEFNCL(0)+30720,6:POKEFNCL(0),102
3272 POKEFNCL(0)+30721,6:POKEFNCL(0),103
3300 PRINT"(HOM)"(2CRSRD)"SPC(5)"(RVSON)"(GRN)"(26PC)"I
3301 PRINT"(HOM)"(2CRSRD)"(RVSON)"(RED)SC="H1"(HOM)"(2CRSRD)"(1CRSRD)"(PUR)SC="H2
3302 PRINT"(HOM)"(2CRSRD)"SPC(5)"(RVSON)"(GRN)VIENTO"INT(ABS(W*0.1))"KMHC(BLK)":RE
TURN

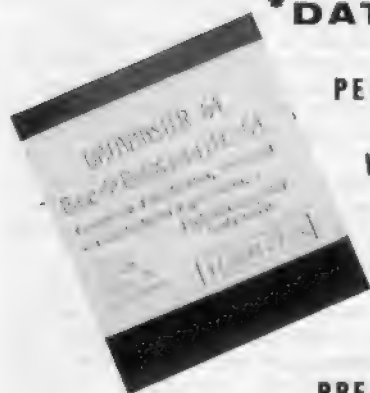
```

Elektrocomputer

ELEKTROCOMPUTER · PRESENTA SUS NUEVOS PRODUCTOS PARA EL VIC-20 Y EL COMMODORE-64. **DATAMASTER 64** Y **CONTROLADOR · C8**, QUE AMPLIAN LAS POSIBILIDADES DE SU ORDENADOR.

DE VENTA EN DISTRIBUIDORES AUTORIZADOS DE TODA ESPAÑA.



***DATAMASTER 64** _ SOFISTICADA BASE DE DATOS PARA EL C-64.

PENSADA PARA TRABAJAR CON LA UNIDAD DE DISCO 1541. SIENDO MUY

VERSATIL APROVECHA AL MAXIMO LA CAPACIDAD DE MANIOBRA Y ALMA-

CENAMIENTO. NUMERO DE REGISTROS VARIABLE *EJ. 5000 DE 30 CA-

RACTERES*. FORMATEADOS Y COPIAS PROGRAMADAS. SALIDA A IM-

PRESORA (PARALELO CENTRONICS Y SERIE RS232) CHEQUEO OPERACIONES

DISCO. GARANTIA 3 MESES. MANUAL COMPL. EN CASTELLANO — P.V.P. 11.800' PTAS.

***CONTROLADOR · C8** _ CONTROLADOR DE 8 RELES

PARA EL VIC-20 Y EL C-64. DE FORMA MUY SENCILLA PODE-

MOS HACER HASTA 255 COMBINACIONES ENTRE LOS 8 RE-

LES, CON UN CONSUMO DE 1000 W. A 220 VOLT. CADA UNO.

CON LO CUAL PODEMOS ACCIONAR TODO TIPO DE LUCES O

MECANISMOS. INSTRUC. INCLUIDAS. 3 MESES GARANTIA

— P.V.P. 9.800' PTAS.



VIA AUGUSTA - 120 - TEF. (93) 2180699 - BARCELONA - 6

VENTANA CBM (IV)

Los capítulos anteriores fueron publicados en Commodore Club. Aquellos lectores que los deseen, les rogamos nos lo comuniquen para poder enviarles copia de dichos artículos.



READY (del chip a la base de datos) (I)

Cumpliendo con lo prometido en números anteriores, voy a comentaros en éste algunos aspectos del sistema co-residente MEC/DOS (MicroElectrónica y Control/Disc Operating Sistem).

MEC/DOS es un sistema operativo co-residente con el BASIC y el KERNAL, escrito en assembler por MEC SOFT para los micros de la serie 7XX de COMMODORE.

Por Rafael NAVARRO

Recomendado para
C-64 y C-700

Lo primero que cabe destacar en él es que su nombre no lo define totalmente ya que los criterios considerados para su bautizo fueron más comerciales y estéticos que técnicos; no se trata exactamente de un sistema operativo de disco ya que, por un lado, funciones elementales de sistema, como la copia de discos o su formatación se dejan al sistema base de COMMODORE y, por otro, MEC/DOS abarca también otras áreas, como la edición de pantallas y el cálculo.

Desde este punto de vista, podría decirse que MEC/DOS es un conjunto de ayudas a la programación que, sin perder su carácter universal, se enfoca a facilitar la programación y el desarrollo de aplicaciones para gestión y control de actividades en la empresa. Acordes con la filosofía de COMMODORE, hemos integrado el sistema en un cartucho con 16K de eeprom, conteniendo el código y 8K de ram de uso interno, evitando de este modo los tiempos de carga (bootstraps) que otros sistemas padecen. Se trata pues de un concepto nuevo entre los productos desarrollados tanto para COMMODORE como para otros sistemas:

- no ocupa RAM del usuario.
- no interfiere el BASIC residente aunque aprovecha algunas de sus rutinas
- debido a su alto nivel de estructuración interna es independiente del lenguaje empleado
- para cuando se trabaje con el BASIC residente (interpretado) dispone de su propio interpretador que trabaja en base a comandos con nombre propio (nada de SYS 34567 o CALL 56678 totalmente carentes de significado) y opera en entrada/salida con parámetros (variables) bajo el mismo formato que el BASIC

- es extremadamente fácil de entender, aprender y utilizar
- dispone de un sistema automático de protección contra errores tanto de usuario como de sistema, lo cual le confiere un alto nivel de seguridad e información, ya que no sólo lo avisa de los estados de error si no que informa de las características de los mismos mediante completos mensajes.

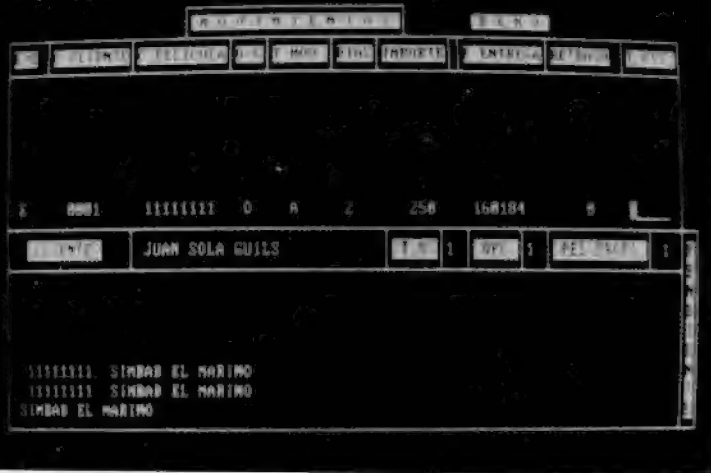
MEC/DOS abarca las siguientes áreas de trabajo:

- Tratamiento de ficheros relativos.
- Tratamiento de ficheros de claves.
- Tratamiento de ficheros tablas.
- Tratamiento de campos (des/empaquetado)
- Edición y tratamiento de pantallas.
- Múltiple precisión.
- Intérprete MEC/DOS (con BASIC residente).

Al final de este artículo se encuentra la tabla resumen de los comandos que acompaña al manual de MEC/DOS. Considérese que en ella aparecen los comandos por el nombre que es inteligible para el interpretador MEC/DOS. Cuando se opere bajo un entorno que no sea el interpretador BASIC de COMMODORE (compilador BASIC, —PET SPEED—, PASCAL —DTL—, u otros) el formato de utilización será diferente, pero siempre posible.

El interpretador

El interpretador está constituido por una serie de programas que comunican al programador con el núcleo de MEC/DOS. Tanto desde el punto de vista interno como en su relación con el programador, MEC/DOS utiliza un buffer



Base de datos



Base de datos

(Fotos cedidas por DENO)

(área de memoria reservada) para comunicar datos de entrada/salida entre las diferentes rutinas que constituyen el sistema y entre éstas y el programador. La función del interpretador es facilitar dicha comunicación a través de variables BASIC. De modo sucinto, podemos definir el siguiente **modus operandi** para el interpretador:

- lectura del comando
- comprobación de la sintaxis
- lectura de parámetros y formatación de los mismos
- ejecución del comando propiamente dicho
- recogida de los datos de salida
- formatación de los mismos
- devolución de resultados a través de variables BASIC
- actualización de la variable OK (avisos y errores)
- impresión de mensaje de error y listado de la línea donde se produjo (opcional)

MEC/DOS ofrece 50 comandos diferentes cuya longitud es de 6 caracteres reconocibles, tras los cuales puede añadirse cualquier afijo para hacerlos más significativos todavía. Por ejemplo, el comando "LIBERA" (liberación de un registro de fichero relativo para su posterior utilización tras una baja) puede ser escrito como "LIBERA EL REGISTRO QUE A CONTINUACION TE INDICO" siendo el aspecto final de la instrucción

algo así como: "LIBERA EL REGISTRO QUE TE INDICO", NF, NR, para liberar al registro NR del fichero NF. Como se ve, no puede ser más claro. En este caso, solamente existen dos parámetros de entrada (NF y NR) y ninguno de salida y obviamos que OK —que será cargado con un 0 si la operación se ha llevado a cabo sin error y con otro valor en caso contrario— no es un parámetro de salida si no la variable de estado de MEC/DOS. El sistema permite la utilización de los siguientes parámetros.

En entrada:

- constantes numéricas
- constantes alfa
- variables numéricas simples o array
- variables alfa simples o array
- cualquier tipo de expresión alfa o numérica.

En salida:

- variables numéricas simples o array
- variables alfa simples o array

Aunque es imposible en este espacio dar una idea exacta de lo que representa MEC/DOS en todos los aspectos, las siguientes líneas pretenden dar una idea muy general de sus posibilidades.

Ficheros relativos

Los ficheros relativos son ficheros de estructura muy similar a los del BASIC de COMMODORE y otros de los basics que ruedan bajo CP/M o MS/DOS. De

hecho, tanto los RELATIVOS como las CLAVES DE MEC/DOS están soportados por ficheros de estructura RELATIVO de COMMODORE. Las diferencias fundamentales estriban en los siguientes puntos.

- MEC/DOS permite tener abiertos 12 ficheros simultáneos
- se efectúa un control de los registros libres y ocupados
- se gestionan automáticamente los campos de cada registro procediendo a su empaquetado y desempaquetado en los procesos de entrada/salida.
- a tal efecto no es necesaria ninguna preparación previa en el programa (existe un descriptor, generado en el momento de definir el fichero). Basta con definir los ficheros una sola vez mediante un utilitario y MEC/DOS recuerda siempre las características de los mismos
- existe una detección automática de errores que no obliga al testado constante de variables de errores (DS y DSS en COMMODORE, ER, EL, ERS en otros sistemas)
- en el capítulo de seguridad, cabe destacar además, la existencia de comandos de repaso de seguridad y arreglo automático de posibles errores.
- grandes facilidades para la ampliación y/o cambio de estructura de los ficheros.

TABLA RESUMEN DE COMANDOS DEL MEC/DOS

INICIALIZACION

COMANDO	EXPLICACION	VALORES DE OK
INIT	Inicializa MEC/DOS reseteando las unidades de disco conectadas. Pone como periférico por defecto de unidad de discos 8.	OK = 0; correcto
DISK,PE	Cambia el número de periférico por defecto para los comandos CHARGE y ASSIGN.	OK = 0; correcto

FICHEROS (GENERAL)

COMANDO	EXPLICACION	VALORES DE OK
ASSIGN, nf, nf\$ [.d(dr)] [.u(pe)] [.P]	Asigna el fichero nf permitiendo al operador trabajar refiriéndose solamente a nf.	OK = 0; correcto OK = 1; comando no permitido OK = 2; nf ya asignado OK = 3; demasiados ficheros asignados OK = 4; demasiados ficheros prioritarios OK = 13; error de checksum en carga de tablas.
KEEP,nf	Para ficheros de tablas graba la tabla y reabre el fichero. Para ficheros relativos graba el descriptor si ha cambiado el primer registro libre y reabre el fichero. En ficheros de claves, cierra y abre el fichero.	OK = 0; correcto OK = 5; fichero no asignado
LOCK,nf [nf,nf...]	Cierra ficheros y graba, si procede, descriptores y/o tablas.	OK = 0; correcto OK = 5; fichero no asignado

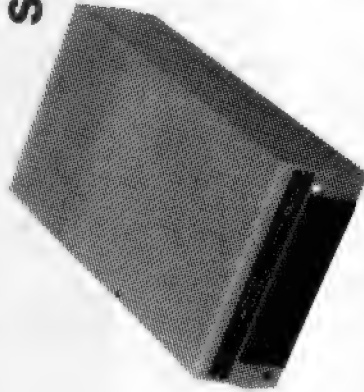
FICHEROS RELATIVOS

COMANDO	EXPLICACION	VALORES DE OK
USE,NF,NR	Coloca el registro NR del fichero NF en estado de "ocupado", permitiendo de este modo, grabar (WRITE) sobre el mismo.	OK = 0; correcto OK = 5; NF no asignado OK = 6; NR fuera de rango OK = 9; NR ya ocupado
LIBERATE,NF,NR	Libera el registro NR del fichero NF.	OK = 0; correcto OK = 1; comando no apropiado OK = 5; NF no asignado OK = 6; NR fuera de rango OK = 7; registro ya liberado
READ,NF,NR	Carga en el buffer el registro NR del fichero NF, dando acceso a consultas y variaciones del contenido de NR.	OK = 0; correcto OK = 5; NF no asignado OK = 6; NR fuera de rango OK = 7; registro libre
WRITE, NF,NR	Graba el contenido del buffer del fichero NF sobre el registro NR del mismo. El contenido del buffer permanece intacto. El registro debe estar ya ocupado.	OK = 0; correcto OK = 5; NF no asignado OK = 6; NR fuera de rango OK = 7; registro libre
AVAIL,NF,NR	Devuelve en OK la disponibilidad del registro NR del fichero NF. Los valores de OK 7 y 9 no producen mensaje en este comando, por meramente informativos.	OK = 5; NF no asignado OK = 6; (NR fuera de rango) OK = 7; registro libre OK = 9; registro ocupado
FIRST,NF,VR	Devuelve en VR el valor del primer registro libre del fichero NF. Cero si NF esta lleno.	OK = 0; correcto OK = 5; NF no asignado OK = 8; NF completo
RELINK,NF	Repasa, caso de existir sospechas sobre su integridad, las cadenas de control de registros libres del fichero NF.	OK = 0; correcto OK = 1; comando no apropiado

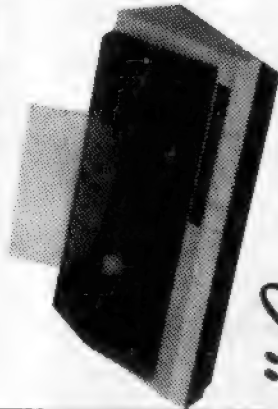
Programas standard medida para

**y "a medida" para
equipos C=commodore**

SISTEMA 8000



COMMODORE 64	79.900 pts.
IMPRESORA GP-100 VC	95.000 pts.
UNIDAD DE DISCOS 1.541	54.900 pts.
PRG. CONTABILIDAD	25.000 pts.
PRG. GESTION EMPRESA	25.000 pts.
PRG. ETIQUETAS	5.000 pts.
TOTAL	284.800 pts.



¡¡Gratis!! SUSCRIPCION POR 1 AÑO
A LA REVISTA **Commodore**

Commodore WORLD

- CONTABILIDAD (10MB)
- GESTION COMER.
- 9000 ARTICULOS
- GEST. INTEGRADA
- ALMACEN
- NOMINAS
- CONTABILIDAD
- GESTION COMERC.
- STOCK ALMACENES
- VIDEO CLUB
- ENTRAPUNT
- QUINIELAS
- AGENDA
- COTIZACION BOLSA
- ETIQUETAS
- PELUQUERIAS
- ETC.

Apellidos
Nombre
Dirección
Población
D. P.

Téléphone

EQUIPO	IMPORTE
OFERTA C 64	244.400
FORMA DE PAGO	TOTAL
TALON BANCARIO REGISTRADO	
CONTRA-REEMBOLSO	
	TOTAL

PEDIÇOS A-

THE NEW YORK PUBLIC LIBRARY

Avenida Cesar Augusto. 72
Telefonos 235682 y 226544
ZARAGOZA-3



¡Arriba Periscopio!

Los gráficos, la generación de sonido, las comunicaciones y un número de funciones de la configuración de un equipo dependen de que el usuario seleccione unos bits específicos de una posición definida de memoria, que haga un "PEEK" en dicha posición para saber cuáles son los bits puestos a 1 ó a 0 y luego haga un "POKE" en esa posición de memoria para poner a 1 o a 0 los bits que necesite.

Para nosotros, quienes a veces tenemos problemas para pensar en valores hexadecimales y binarios e ignoramos cuáles son los valores almacenados en la memoria, el programa "Periscopio" para el Commodore 64 y el VIC-20 proporciona un instrumento útil para que entendamos mejor nuestras máquinas.

El programa "Periscopio" se escribe totalmente en Basic y permite que el usuario vuelque los valores hexadecimales almacenados en una serie de posiciones de memoria, corregir (poner un PATCH) (en hexadecimal) unos valores en una serie de posiciones de memoria, desensamblar los valores hexadecimales de una serie de posiciones de memoria en instrucciones de lenguaje ensamblador del 6502/6510, y convertir los valores de números de decimal, hexadecimal o binario en todas las tres bases.

Operación del Programa.

Cuando se ejecuta el programa "Periscopio", lo primero que hace es borrar la pantalla y representar un menú de opciones. Se selecciona una opción mediante la pulsación de un número de 1 a 5 antes de pulsar la tecla de entrar (ENTER). Cuando se haya seleccionado una opción correcta, el programa pregunta o bien una dirección de principio (¿Sí?) o bien una pregunta de conversión de base (13=DEC 2=HEX 3=BIN)?

Vamos a estudiar las opciones para ver lo que pueden hacer para nosotros.

Opción 1: Volcado (Dump) - La opción de volcado representa los valores almacenados en la memoria en notación hexadecimal (dos dígitos hexadecimales para cada byte de memoria). Los valores de memoria se representan a partir de una dirección inicial a una dirección final, introducidos mediante el teclado. Cuando se selecciona la opción, se representa una pregunta de dirección de principio (¿Sí?)

A continuación el programa espera que se introduzca una dirección de memoria de dos bytes (cuatro dígitos hexadecimales). Si se produce un error al introducir la dirección, no se puede corregir mediante una operación de retroceso (backspace), dado que el programa no reconoce INST/DEL. Sin embargo, los datos de la dirección se pueden volver a teclear justo después del error en la misma línea. El programa acepta los últimos cuatro dígitos tecleados antes de haber pulsado la tecla "RETURN", sea cual sea el número total de caracteres que hayan sido tecleados en la línea.

Después de introducir la dirección inicial, el programa contesta con un mensaje de dirección final de >. La dirección

final se introduce de la misma forma que la dirección inicial, considerando los últimos cuatro dígitos tecleados como el valor real.

Si se introducen solamente dos o tres dígitos hexadecimales, el programa supone que los caracteres de la extrema izquierda son ceros. Por ejemplo, si se teclaea AE hexadecimal y se pulsa la tecla "RETURN", el programa supone que significa la dirección de OOE hexadecimal.

Si el programa tiene las direcciones de principio y final para el volcado de memoria, se presenta un mensaje de "Salida de Impresora (Si/No)". Una respuesta de Si dirige la salida a la impresora 1525 de Commodore. Si se introduce una N, o simplemente se pulsa la tecla "RETURN", la salida se dirige a la pantalla.

Si se pulsa la tecla "ENTER", el programa imprime una dirección de memoria hexadecimal seguida por ocho valores hexadecimales almacenados en la memoria del VIC-20. El programa sigue imprimiendo las direcciones de memoria, seguidas por los valores de memoria en grupos de seis u ocho bytes, hasta que se llega a la dirección final o hasta que se pulsa cualquier tecla alfanumérica en el teclado cuando se representan los valores en la pantalla.

Cuando se pulsa una tecla durante una operación de volcado en la pantalla, la salida se detiene y el programa espera que se pulse otra tecla antes de continuar la operación. Así los datos no se pierden fuera de la pantalla antes de que se puedan leer. Si la tecla que se pulsa para interrumpir la representación es la "X", la operación de volcado finaliza y el programa vuelve a presentar la pregunta ¿Yes? (¿Sí?). Si la tecla "X" se vuelve a pulsar, el programa vuelve al menú.

Si se introduce la dirección hexadecimal, el programa pedirá la dirección final y la salida de impresora, y luego realiza una operación de volcado de los valores de memoria empezando por la nueva dirección inicial. Cuando se llega a la dirección final, el programa representa la pregunta ¿Yes? y otro volcado puede ser seleccionado, o si no, se pulsa la tecla "X" para volver al menú.

Opción 2: Corrección (Patch) - La opción de corrección (Patch) permite introducir valores hexadecimales en la memoria. Cuando la opción es seleccionada, el programa presenta la segunda Yes? Una dirección inicial debe introducirse de la misma manera en que se hizo con la opción de volcado, o si no, se puede pulsar la tecla "X" para volver al menú.

Después de introducir una dirección inicial, el programa presenta el mensaje >.

COMMODORE 64

o
VIC-20 de
3 K o más

Se introduce la dirección final o una X para volver al menú. Después de introducir la dirección final, el programa presenta la dirección inicial en hexadecimal y espera que se introduzca un valor hexadecimal de un byte. Un error al teclear se puede corregir de la misma forma en que se hace cuando se introduce una dirección; se teclaea el valor correcto justo después del error. El programa acepta los últimos dos dígitos hexadecimales en la línea como el valor correcto que se coloca en la memoria.

El programa continuará representando la siguiente posición de memoria y esperará que el usuario entre un nuevo valor. Esta secuencia continuará hasta que la posición final sea alcanzada o hasta que se pulse la tecla "X". Si la tecla "X" es pulsada, la rutina de entrada de datos terminará y aparecerá el mensaje "¿Si?". Lo mismo que en el caso de la función de volcado, un nuevo juego de direcciones pueden entrarse, o pulsando la tecla "X" otra vez podemos volver al menú.

Opción 3: Desensamblado - La operación de desensamblado presentará una sección de memoria como instrucciones del ensamblador del 6502/6510. La rutina traducirá los valores hexadecimales en una posición de memoria particular de 6502/6510 en mnemónicos del lenguaje "assembler" del 6502/6510 para una instrucción particular, y la presenta en pantalla. La rutina a continuación presentará los valores de memoria de la siguiente o de las siguientes dos posiciones, si esto lo requiere el direccionamiento del programa en lenguaje máquina.

Los mensajes de la opción desensamblador y la entrada de datos son exactamente iguales que en el caso de la opción de volcado. Se presentan los mensajes de entrada y final de las direcciones y el mensaje de salida e impresora. En ambos mensajes, la tecla "X" puede ser pulsada para volver al menú. Mientras que el desensamblaje se presenta en la pantalla, la rutina puede ser interrumpida de la misma forma que la rutina de volcado. Se puede especificar una nueva sección de memoria o volver al menú.

Opción 4: Conversión - La opción de conversión puede convertir un valor numérico entrado como decimal, hexadecimal o binario en cualquiera de las dos bases restantes y su valor presentado en pantalla. Cuando se selecciona esta opción, el programa presentará el mensaje de conversión (1=DEC 2=HEX 3=BIN?). A continuación, hay que pulsar un número correspondiente a la base que se desee a bien pulsar la tecla "X" para volver al menú. Después de haber seleccionado la base, el programa presentará el mensaje DEC?, HEX? o BIN?. A continuación el programa esperará la entrada del valor en la base especificada.

Los números decimales tienen que estar comprendidos entre 0 y 65535 sin ningún espacio entre los dígitos. Los números hexadecimales pueden tener hasta cuatro caracteres de longitud y tienen que intro-

ducirse de la misma forma que las entradas de direcciones. Los números binarios tienen que tener desde uno, hasta máximo 16 dígitos, formados por unos y ceros. De la misma forma que con los números decimales, es un error introducir espacios en blanco entre los distintos dígitos. Cuando se estén introduciendo números decimales o binarios, se puede utilizar la tecla de "borrado" para volver a la posición anterior y corregir la última entrada antes de pulsar la tecla "ENTER".

Cuando un valor ha sido introducido, el programa presentará su valor convertido en una de las otras dos bases soportadas. Un número decimal se representará como un valor entero. Un número hexadecimal se presentará como cuatro caracteres hexadecimales. Un número binario se representará como dos grupos de ocho dígitos (que es la representación binaria de dos bytes).

Después de haber presentado los valores convertidos, el programa presentará el mensaje (1=DEC 2=HEX 3=BIN)? y otro valor de base puede ser introducido, o se puede pulsar la tecla "X" para volver al menú.

Opción 5: Fin - La opción fin comprueba para asegurarse de que la impresora está desconectada y que el programa vuelve a dar control al editor Basic de pantalla del Commodore.

Descripción del Programa

El programa "Periscopio" ha sido diseñado para ofrecer un instrumento útil y al mismo tiempo utilizar la mayor memoria posible. Con esto en mente, en cada línea se han escrito el mayor número posible de instrucciones. No se han utilizado las instrucciones REM por el mismo motivo. Cuando se introduce el programa, las líneas 600-610 tienen que ser descartadas para ahorrar espacio. El programa ocupará entonces aproximadamente 3000 bytes de memoria y aproximadamente 3785 bytes de memoria cuando funcione.

El programa puede ser condensado posteriormente usando los comandos abreviados de dos teclas soportados por el Basic de Commodore. Por ejemplo, las líneas 611 y 615 pueden ser combinadas. Los comandos de datos de las líneas 783-798 pueden ser condensados en menos líneas en total. El programa ha sido escrito tal cual para facilitar la entrada del listado dado que las líneas han sido construidas con los comandos abreviados de dos teclas que normalmente exceden de los 80 caracteres en el Commodore 64 y de los 88 caracteres en el VIC-20 cuando se listan en pantalla. Aunque las líneas traducidas entren dentro de una línea de Basic en memoria, esta línea no puede ser editada en la pantalla sin perder datos, de forma que la línea tiene que ser escrita de nuevo desde cero para corregir los errores. El programa tiene que ser escrito como en el listado, hay que ejecutarlo para detectar errores, y corregirlo. Una vez que funcione correctamente, puede ser condensado.



El listado del programa puede ser utilizado tal cual en el Commodore 64. Hay que cambiar una línea para ejecutar el programa en el VIC-20. En la línea 900, el valor (P-S)/8 aparece dos veces. Cambiar ocho por seis en ambos sitios, y la opción de volcado entrará en la anchura del VIC-20.

En un VIC-20 no ampliado, el usuario podría querer borrar algunas opciones para tener más memoria libre cuando se ejecuta el programa "Periscopio". Por ejemplo, borrando las líneas 670-696 y borrando el valor 670 desde la línea 655 y PRINT"4) CONVERT:" desde 640 y cambiando PRINT"5) END en PRINT"4) END", la opción entera de conversión será eliminada del programa. Como herramienta para modificaciones posteriores del programa, a continuación tenemos unas descripciones de las rutinas.

Líneas 611-620: Esta rutina inicializa y llena dos "arrays" con datos. El "array" 0% contiene los valores empaquetados que, cuando son desempaquetados, nos dan el puntero del "array" M\$ y el puntero de las subrutinas correspondientes para imprimir los diferentes modos del direccionamiento del 6502/6510.

Líneas 630-659: Estas líneas contienen las rutinas del programa principal, que presentan el menú de opciones en las líneas 630 y 640, hacen un "input" de la opción seleccionada en la línea 645 y resetean la impresora y las direcciones de las entradas de las subrutinas en la línea 650. La línea 655 dirige el programa a la subrutina de la opción seleccionada.

Líneas 665-667: Esta rutina es llamada para inhibir la salida a la impresora.

Línea 660: Esta línea llama a la rutina que inhibe la impresora y a continuación acaba el programa.

Líneas 670-696: La opción de rutina de conversión imprime los mensajes de conversión en la línea 670. Las líneas 672-682 tienen las instrucciones de entrada de selección de la base, e imprimen el número de bases y llaman a las subrutinas de conversión. Las líneas 684-686 convierten una entrada binaria en un número decimal y luego llaman a la rutina de impresión del número decimal en la línea 688.

Un número decimal es convertido en binario, y luego impreso como un número binario de 16 bits en las líneas 690-696. La línea 696 llama una subrutina en la línea 950; esta subrutina convierte un número

Listado 1

UP PERISCOPE

```

600 REM*****
601 REM*
602 REM*      N PERISCOPE N
603 REM* FOR COMMODORE-64 AND VIC 20
604 REM*
605 REM*
606 REM*
607 REM*
608 REM*
609 REM*
610 REM*****
611 DIM#(56),O%(255):FORP=0TO56:READM(P):NEXT:FORP=0TO255:READW
612 IFW<99THENW=W*100
613 O%(P)=W:NEXT
614 PRINTCHR$(147):R=0:PRINTCHR$(18):"(SPC)PERISCOPIO(SPC)ABUI"(SPC)*:PRINT:PR
INT":(SPC)VOLCADO"
615 PRINT"2):(SPC)CORRECCION":PRINT"3):(SPC)DESENSAMBLADO":PRINT"4):(SPC)CONVERSION
":PRINT"5):(SPC)FIN"
616 PRINT:INPUT"OPCION(SPC)1:IC:IFC)3ORC)1THEN655
617 GOSUB665:GOSUB970:IFRTHEN630
618 ONCGOSUB900,800,700,670,660:R=0:IFC)3THEN630
619 GOTO630
620 :GOSUB665:END
621 IFL=1THENPRINT#4:CLOSE4:L=0:P$=""
622 RETURN
623 R=0:PRINT"1:DEC(SPC)2=HEX(SPC)3=BIN:(SPC)7"
624 GETK$:IFK$=""THEN672
625 IFK$="X"THENRETURN
626 IFK$="1"THENPRINT"DEC":E=0:INPUT:P=E:GOSUB696:GOSUB690
627 IFK$="3"THENPRINT"BIN":GOSUB684:GOSUB696
628 IFK$="2"THENPRINT"HEX":GOSUB980:IFR)1THENGOSUB688:GOSUB690
629 PRINT:GOTO670
630 E=0:O=0:K$="":INPUTK$:FORX=LEN(K$)TO1STEP-1:O=O+1
631 E=E+VAL(MID$(K$,O,1))*2*(X-1):NEXTX:GOSUB688:RETURN
632 PRINT"DEC":E:RETURN
633 PRINT"BIN":FORX=1TO1STEP-1:O=0:IFE-2)X=0THENE=2)X:O=1
634 IFX=7THENPRINT"(SPC)"
635 PRINTRIGHT$(STR$(O),1):NEXTX:PRINTRIGHT$(STR$(E),1):RETURN
636 P=E:PRINT"HEX":GOSUB950:PRINT:RETURN
637 P=E:NEXTP:PRINT:RETURN
638 GOSUB990:FORP=0TO56:IFLTHEN710
639 GOSUB995:IFRTHEN698
640 GOSUB950:W=0:(PEEK(P)):Z=INT(W/100):W=W-Z*100:PRINT"(SPC)1"IM(Z):"(SPC)"
641 ONWGOSUB725,700,735,740,782,750,755,760,765,770,775,775:PRINT:PRINT:RETURN
642 PRINT"W":GOTO700
643 GOSUB700:PRINT",X":RETURN
644 GOSUB700:PRINT",Y":RETURN
645 GOSUB762:PRINT",X":RETURN
646 GOSUB762:PRINT",Y":RETURN
647 GOSUB782:PRINT",Y":RETURN
648 P=F+1:O=F:H=PEEK(P):IFH)127THENP=P-256
649 F=P+1:H=PRINT"X":GOSUB950:P=O:RETURN
650 PRINT"1:GOSUB700:PRINT",X":RETURN
651 PRINT"1:GOSUB700:PRINT",Y":RETURN
652 PRINT"1:GOSUB700:PRINT",Y":RETURN
653 PRINT"1:GOSUB700:PRINT",Y":RETURN
654 PRINT"1:GOSUB700:PRINT",Y":RETURN
655 PRINT"1:GOSUB700:PRINT",Y":RETURN
656 PRINT"1:GOSUB700:PRINT",Y":RETURN
657 PRINT"1:GOSUB700:PRINT",Y":RETURN
658 PRINT"1:GOSUB700:PRINT",Y":RETURN
659 PRINT"1:GOSUB700:PRINT",Y":RETURN
660 PRINT"1:GOSUB700:PRINT",Y":RETURN
661 PRINT"1:GOSUB700:PRINT",Y":RETURN
662 PRINT"1:GOSUB700:PRINT",Y":RETURN
663 PRINT"1:GOSUB700:PRINT",Y":RETURN
664 PRINT"1:GOSUB700:PRINT",Y":RETURN
665 PRINT"1:GOSUB700:PRINT",Y":RETURN
666 PRINT"1:GOSUB700:PRINT",Y":RETURN
667 PRINT"1:GOSUB700:PRINT",Y":RETURN
668 PRINT"1:GOSUB700:PRINT",Y":RETURN
669 PRINT"1:GOSUB700:PRINT",Y":RETURN
670 PRINT"1:GOSUB700:PRINT",Y":RETURN
671 PRINT"1:GOSUB700:PRINT",Y":RETURN
672 PRINT"1:GOSUB700:PRINT",Y":RETURN
673 PRINT"1:GOSUB700:PRINT",Y":RETURN
674 PRINT"1:GOSUB700:PRINT",Y":RETURN
675 PRINT"1:GOSUB700:PRINT",Y":RETURN
676 PRINT"1:GOSUB700:PRINT",Y":RETURN
677 PRINT"1:GOSUB700:PRINT",Y":RETURN
678 PRINT"1:GOSUB700:PRINT",Y":RETURN
679 PRINT"1:GOSUB700:PRINT",Y":RETURN
680 PRINT"1:GOSUB700:PRINT",Y":RETURN
681 PRINT"1:GOSUB700:PRINT",Y":RETURN
682 PRINT"1:GOSUB700:PRINT",Y":RETURN
683 PRINT"1:GOSUB700:PRINT",Y":RETURN
684 PRINT"1:GOSUB700:PRINT",Y":RETURN
685 PRINT"1:GOSUB700:PRINT",Y":RETURN
686 PRINT"1:GOSUB700:PRINT",Y":RETURN
687 PRINT"1:GOSUB700:PRINT",Y":RETURN
688 PRINT"1:GOSUB700:PRINT",Y":RETURN
689 PRINT"1:GOSUB700:PRINT",Y":RETURN
690 PRINT"1:GOSUB700:PRINT",Y":RETURN
691 PRINT"1:GOSUB700:PRINT",Y":RETURN
692 PRINT"1:GOSUB700:PRINT",Y":RETURN
693 PRINT"1:GOSUB700:PRINT",Y":RETURN
694 PRINT"1:GOSUB700:PRINT",Y":RETURN
695 PRINT"1:GOSUB700:PRINT",Y":RETURN
696 PRINT"1:GOSUB700:PRINT",Y":RETURN
697 PRINT"1:GOSUB700:PRINT",Y":RETURN
698 PRINT"1:GOSUB700:PRINT",Y":RETURN
699 PRINT"1:GOSUB700:PRINT",Y":RETURN
700 PRINT"1:GOSUB700:PRINT",Y":RETURN
701 PRINT"1:GOSUB700:PRINT",Y":RETURN
702 PRINT"1:GOSUB700:PRINT",Y":RETURN
703 PRINT"1:GOSUB700:PRINT",Y":RETURN
704 PRINT"1:GOSUB700:PRINT",Y":RETURN
705 PRINT"1:GOSUB700:PRINT",Y":RETURN
706 PRINT"1:GOSUB700:PRINT",Y":RETURN
707 PRINT"1:GOSUB700:PRINT",Y":RETURN
708 PRINT"1:GOSUB700:PRINT",Y":RETURN
709 PRINT"1:GOSUB700:PRINT",Y":RETURN
710 PRINT"1:GOSUB700:PRINT",Y":RETURN
711 PRINT"1:GOSUB700:PRINT",Y":RETURN
712 PRINT"1:GOSUB700:PRINT",Y":RETURN
713 PRINT"1:GOSUB700:PRINT",Y":RETURN
714 PRINT"1:GOSUB700:PRINT",Y":RETURN
715 PRINT"1:GOSUB700:PRINT",Y":RETURN
716 PRINT"1:GOSUB700:PRINT",Y":RETURN
717 PRINT"1:GOSUB700:PRINT",Y":RETURN
718 PRINT"1:GOSUB700:PRINT",Y":RETURN
719 PRINT"1:GOSUB700:PRINT",Y":RETURN
720 PRINT"1:GOSUB700:PRINT",Y":RETURN
721 PRINT"1:GOSUB700:PRINT",Y":RETURN
722 PRINT"1:GOSUB700:PRINT",Y":RETURN
723 PRINT"1:GOSUB700:PRINT",Y":RETURN
724 PRINT"1:GOSUB700:PRINT",Y":RETURN
725 PRINT"1:GOSUB700:PRINT",Y":RETURN
726 PRINT"1:GOSUB700:PRINT",Y":RETURN
727 PRINT"1:GOSUB700:PRINT",Y":RETURN
728 PRINT"1:GOSUB700:PRINT",Y":RETURN
729 PRINT"1:GOSUB700:PRINT",Y":RETURN
730 PRINT"1:GOSUB700:PRINT",Y":RETURN
731 PRINT"1:GOSUB700:PRINT",Y":RETURN
732 PRINT"1:GOSUB700:PRINT",Y":RETURN
733 PRINT"1:GOSUB700:PRINT",Y":RETURN
734 PRINT"1:GOSUB700:PRINT",Y":RETURN
735 PRINT"1:GOSUB700:PRINT",Y":RETURN
736 PRINT"1:GOSUB700:PRINT",Y":RETURN
737 PRINT"1:GOSUB700:PRINT",Y":RETURN
738 PRINT"1:GOSUB700:PRINT",Y":RETURN
739 PRINT"1:GOSUB700:PRINT",Y":RETURN
740 PRINT"1:GOSUB700:PRINT",Y":RETURN
741 PRINT"1:GOSUB700:PRINT",Y":RETURN
742 PRINT"1:GOSUB700:PRINT",Y":RETURN
743 PRINT"1:GOSUB700:PRINT",Y":RETURN
744 PRINT"1:GOSUB700:PRINT",Y":RETURN
745 PRINT"1:GOSUB700:PRINT",Y":RETURN
746 PRINT"1:GOSUB700:PRINT",Y":RETURN
747 PRINT"1:GOSUB700:PRINT",Y":RETURN
748 PRINT"1:GOSUB700:PRINT",Y":RETURN
749 PRINT"1:GOSUB700:PRINT",Y":RETURN
750 PRINT"1:GOSUB700:PRINT",Y":RETURN
751 PRINT"1:GOSUB700:PRINT",Y":RETURN
752 PRINT"1:GOSUB700:PRINT",Y":RETURN
753 PRINT"1:GOSUB700:PRINT",Y":RETURN
754 PRINT"1:GOSUB700:PRINT",Y":RETURN
755 PRINT"1:GOSUB700:PRINT",Y":RETURN
756 PRINT"1:GOSUB700:PRINT",Y":RETURN
757 PRINT"1:GOSUB700:PRINT",Y":RETURN
758 PRINT"1:GOSUB700:PRINT",Y":RETURN
759 PRINT"1:GOSUB700:PRINT",Y":RETURN
760 PRINT"1:GOSUB700:PRINT",Y":RETURN
761 PRINT"1:GOSUB700:PRINT",Y":RETURN
762 PRINT"1:GOSUB700:PRINT",Y":RETURN
763 PRINT"1:GOSUB700:PRINT",Y":RETURN
764 PRINT"1:GOSUB700:PRINT",Y":RETURN
765 PRINT"1:GOSUB700:PRINT",Y":RETURN
766 PRINT"1:GOSUB700:PRINT",Y":RETURN
767 PRINT"1:GOSUB700:PRINT",Y":RETURN
768 PRINT"1:GOSUB700:PRINT",Y":RETURN
769 PRINT"1:GOSUB700:PRINT",Y":RETURN
770 PRINT"1:GOSUB700:PRINT",Y":RETURN
771 PRINT"1:GOSUB700:PRINT",Y":RETURN
772 PRINT"1:GOSUB700:PRINT",Y":RETURN
773 PRINT"1:GOSUB700:PRINT",Y":RETURN
774 PRINT"1:GOSUB700:PRINT",Y":RETURN
775 PRINT"1:GOSUB700:PRINT",Y":RETURN
776 PRINT"1:GOSUB700:PRINT",Y":RETURN
777 PRINT"1:GOSUB700:PRINT",Y":RETURN
778 PRINT"1:GOSUB700:PRINT",Y":RETURN
779 PRINT"1:GOSUB700:PRINT",Y":RETURN
780 PRINT"1:GOSUB700:PRINT",Y":RETURN
781 PRINT"1:GOSUB700:PRINT",Y":RETURN
782 PRINT"1:GOSUB700:PRINT",Y":RETURN
783 PRINT"1:GOSUB700:PRINT",Y":RETURN
784 PRINT"1:GOSUB700:PRINT",Y":RETURN
785 PRINT"1:GOSUB700:PRINT",Y":RETURN
786 PRINT"1:GOSUB700:PRINT",Y":RETURN
787 PRINT"1:GOSUB700:PRINT",Y":RETURN
788 PRINT"1:GOSUB700:PRINT",Y":RETURN
789 PRINT"1:GOSUB700:PRINT",Y":RETURN
790 PRINT"1:GOSUB700:PRINT",Y":RETURN
791 PRINT"1:GOSUB700:PRINT",Y":RETURN
792 PRINT"1:GOSUB700:PRINT",Y":RETURN
793 PRINT"1:GOSUB700:PRINT",Y":RETURN
794 PRINT"1:GOSUB700:PRINT",Y":RETURN
795 PRINT"1:GOSUB700:PRINT",Y":RETURN
796 PRINT"1:GOSUB700:PRINT",Y":RETURN
797 PRINT"1:GOSUB700:PRINT",Y":RETURN
798 PRINT"1:GOSUB700:PRINT",Y":RETURN
799 PRINT"1:GOSUB700:PRINT",Y":RETURN
800 PRINT"1:GOSUB700:PRINT",Y":RETURN
801 PRINT"1:GOSUB700:PRINT",Y":RETURN
802 PRINT"1:GOSUB700:PRINT",Y":RETURN
803 PRINT"1:GOSUB700:PRINT",Y":RETURN
804 PRINT"1:GOSUB700:PRINT",Y":RETURN
805 PRINT"1:GOSUB700:PRINT",Y":RETURN
806 PRINT"1:GOSUB700:PRINT",Y":RETURN
807 PRINT"1:GOSUB700:PRINT",Y":RETURN
808 PRINT"1:GOSUB700:PRINT",Y":RETURN
809 PRINT"1:GOSUB700:PRINT",Y":RETURN
810 PRINT"1:GOSUB700:PRINT",Y":RETURN
811 PRINT"1:GOSUB700:PRINT",Y":RETURN
812 PRINT"1:GOSUB700:PRINT",Y":RETURN
813 PRINT"1:GOSUB700:PRINT",Y":RETURN
814 PRINT"1:GOSUB700:PRINT",Y":RETURN
815 PRINT"1:GOSUB700:PRINT",Y":RETURN
816 PRINT"1:GOSUB700:PRINT",Y":RETURN
817 PRINT"1:GOSUB700:PRINT",Y":RETURN
818 PRINT"1:GOSUB700:PRINT",Y":RETURN
819 PRINT"1:GOSUB700:PRINT",Y":RETURN
820 PRINT"1:GOSUB700:PRINT",Y":RETURN
821 PRINT"1:GOSUB700:PRINT",Y":RETURN
822 PRINT"1:GOSUB700:PRINT",Y":RETURN
823 PRINT"1:GOSUB700:PRINT",Y":RETURN
824 PRINT"1:GOSUB700:PRINT",Y":RETURN
825 PRINT"1:GOSUB700:PRINT",Y":RETURN
826 PRINT"1:GOSUB700:PRINT",Y":RETURN
827 PRINT"1:GOSUB700:PRINT",Y":RETURN
828 PRINT"1:GOSUB700:PRINT",Y":RETURN
829 PRINT"1:GOSUB700:PRINT",Y":RETURN
830 PRINT"1:GOSUB700:PRINT",Y":RETURN
831 PRINT"1:GOSUB700:PRINT",Y":RETURN
832 PRINT"1:GOSUB700:PRINT",Y":RETURN
833 PRINT"1:GOSUB700:PRINT",Y":RETURN
834 PRINT"1:GOSUB700:PRINT",Y":RETURN
835 PRINT"1:GOSUB700:PRINT",Y":RETURN
836 PRINT"1:GOSUB700:PRINT",Y":RETURN
837 PRINT"1:GOSUB700:PRINT",Y":RETURN
838 PRINT"1:GOSUB700:PRINT",Y":RETURN
839 PRINT"1:GOSUB700:PRINT",Y":RETURN
840 PRINT"1:GOSUB700:PRINT",Y":RETURN
841 PRINT"1:GOSUB700:PRINT",Y":RETURN
842 PRINT"1:GOSUB700:PRINT",Y":RETURN
843 PRINT"1:GOSUB700:PRINT",Y":RETURN
844 PRINT"1:GOSUB700:PRINT",Y":RETURN
845 PRINT"1:GOSUB700:PRINT",Y":RETURN
846 PRINT"1:GOSUB700:PRINT",Y":RETURN
847 PRINT"1:GOSUB700:PRINT",Y":RETURN
848 PRINT"1:GOSUB700:PRINT",Y":RETURN
849 PRINT"1:GOSUB700:PRINT",Y":RETURN
850 PRINT"1:GOSUB700:PRINT",Y":RETURN
851 PRINT"1:GOSUB700:PRINT",Y":RETURN
852 PRINT"1:GOSUB700:PRINT",Y":RETURN
853 PRINT"1:GOSUB700:PRINT",Y":RETURN
854 PRINT"1:GOSUB700:PRINT",Y":RETURN
855 PRINT"1:GOSUB700:PRINT",Y":RETURN
856 PRINT"1:GOSUB700:PRINT",Y":RETURN
857 PRINT"1:GOSUB700:PRINT",Y":RETURN
858 PRINT"1:GOSUB700:PRINT",Y":RETURN
859 PRINT"1:GOSUB700:PRINT",Y":RETURN
860 PRINT"1:GOSUB700:PRINT",Y":RETURN
861 PRINT"1:GOSUB700:PRINT",Y":RETURN
862 PRINT"1:GOSUB700:PRINT",Y":RETURN
863 PRINT"1:GOSUB700:PRINT",Y":RETURN
864 PRINT"1:GOSUB700:PRINT",Y":RETURN
865 PRINT"1:GOSUB700:PRINT",Y":RETURN
866 PRINT"1:GOSUB700:PRINT",Y":RETURN
867 PRINT"1:GOSUB700:PRINT",Y":RETURN
868 PRINT"1:GOSUB700:PRINT",Y":RETURN
869 PRINT"1:GOSUB700:PRINT",Y":RETURN
870 PRINT"1:GOSUB700:PRINT",Y":RETURN
871 PRINT"1:GOSUB700:PRINT",Y":RETURN
872 PRINT"1:GOSUB700:PRINT",Y":RETURN
873 PRINT"1:GOSUB700:PRINT",Y":RETURN
874 PRINT"1:GOSUB700:PRINT",Y":RETURN
875 PRINT"1:GOSUB700:PRINT",Y":RETURN
876 PRINT"1:GOSUB700:PRINT",Y":RETURN
877 PRINT"1:GOSUB700:PRINT",Y":RETURN
878 PRINT"1:GOSUB700:PRINT",Y":RETURN
879 PRINT"1:GOSUB700:PRINT",Y":RETURN
880 PRINT"1:GOSUB700:PRINT",Y":RETURN
881 PRINT"1:GOSUB700:PRINT",Y":RETURN
882 PRINT"1:GOSUB700:PRINT",Y":RETURN
883 PRINT"1:GOSUB700:PRINT",Y":RETURN
884 PRINT"1:GOSUB700:PRINT",Y":RETURN
885 PRINT"1:GOSUB700:PRINT",Y":RETURN
886 PRINT"1:GOSUB700:PRINT",Y":RETURN
887 PRINT"1:GOSUB700:PRINT",Y":RETURN
888 PRINT"1:GOSUB700:PRINT",Y":RETURN
889 PRINT"1:GOSUB700:PRINT",Y":RETURN
890 PRINT"1:GOSUB700:PRINT",Y":RETURN
891 PRINT"1:GOSUB700:PRINT",Y":RETURN
892 PRINT"1:GOSUB700:PRINT",Y":RETURN
893 PRINT"1:GOSUB700:PRINT",Y":RETURN
894 PRINT"1:GOSUB700:PRINT",Y":RETURN
895 PRINT"1:GOSUB700:PRINT",Y":RETURN
896 PRINT"1:GOSUB700:PRINT",Y":RETURN
897 PRINT"1:GOSUB700:PRINT",Y":RETURN
898 PRINT"1:GOSUB700:PRINT",Y":RETURN
899 PRINT"1:GOSUB700:PRINT",Y":RETURN
900 PRINT"1:GOSUB700:PRINT",Y":RETURN
901 PRINT"1:GOSUB700:PRINT",Y":RETURN
902 PRINT"1:GOSUB700:PRINT",Y":RETURN
903 PRINT"1:GOSUB700:PRINT",Y":RETURN
904 PRINT"1:GOSUB700:PRINT",Y":RETURN
905 PRINT"1:GOSUB700:PRINT",Y":RETURN
906 PRINT"1:GOSUB700:PRINT",Y":RETURN
907 PRINT"1:GOSUB700:PRINT",Y":RETURN
908 PRINT"1:GOSUB700:PRINT",Y":RETURN
909 PRINT"1:GOSUB700:PRINT",Y":RETURN
910 PRINT"1:GOSUB700:PRINT",Y":RETURN
911 PRINT"1:GOSUB700:PRINT",Y":RETURN
912 PRINT"1:GOSUB700:PRINT",Y":RETURN
913 PRINT"1:GOSUB700:PRINT",Y":RETURN
914 PRINT"1:GOSUB700:PRINT",Y":RETURN
915 PRINT"1:GOSUB700:PRINT",Y":RETURN
916 PRINT"1:GOSUB700:PRINT",Y":RETURN
917 PRINT"1:GOSUB700:PRINT",Y":RETURN
918 PRINT"1:GOSUB700:PRINT",Y":RETURN
919 PRINT"1:GOSUB700:PRINT",Y":RETURN
920 PRINT"1:GOSUB700:PRINT",Y":RETURN
921 PRINT"1:GOSUB700:PRINT",Y":RETURN
922 PRINT"1:GOSUB700:PRINT",Y":RETURN
923 PRINT"1:GOSUB700:PRINT",Y":RETURN
924 PRINT"1:GOSUB700:PRINT",Y":RETURN
925 PRINT"1:GOSUB700:PRINT",Y":RETURN
926 PRINT"1:GOSUB700:PRINT",Y":RETURN
927 PRINT"1:GOSUB700:PRINT",Y":RETURN
928 PRINT"1:GOSUB700:PRINT",Y":RETURN
929 PRINT"1:GOSUB700:PRINT",Y":RETURN
930 PRINT"1:GOSUB700:PRINT",Y":RETURN
931 PRINT"1:GOSUB700:PRINT",Y":RETURN
932 PRINT"1:GOSUB700:PRINT",Y":RETURN
933 PRINT"1:GOSUB700:PRINT",Y":RETURN
934 PRINT"1:GOSUB700:PRINT",Y":RETURN
935 PRINT"1:GOSUB700:PRINT",Y":RETURN
936 PRINT"1:GOSUB700:PRINT",Y":RETURN
937 PRINT"1:GOSUB700:PRINT",Y":RETURN
938 PRINT"1:GOSUB700:PRINT",Y":RETURN
939 PRINT"1:GOSUB700:PRINT",Y":RETURN
940 PRINT"1:GOSUB700:PRINT",Y":RETURN
941 PRINT"1:GOSUB700:PRINT",Y":RETURN
942 PRINT"1:GOSUB700:PRINT",Y":RETURN
943 PRINT"1:GOSUB700:PRINT",Y":RETURN
944 PRINT"1:GOSUB700:PRINT",Y":RETURN
945 PRINT"1:GOSUB700:PRINT",Y":RETURN
946 PRINT"1:GOSUB700:PRINT",Y":RETURN
947 PRINT"1:GOSUB700:PRINT",Y":RETURN
948 PRINT"1:GOSUB700:PRINT",Y":RETURN
949 PRINT"1:GOSUB700:PRINT",Y":RETURN
950 PRINT"1:GOSUB700:PRINT",Y":RETURN
951 PRINT"1:GOSUB700:PRINT",Y":RETURN
952 PRINT"1:GOSUB700:PRINT",Y":RETURN
953 PRINT"1:GOSUB700:PRINT",Y":RETURN
954 PRINT"1:GOSUB700:PRINT",Y":RETURN
955 PRINT"1:GOSUB700:PRINT",Y":RETURN
956 PRINT"1:GOSUB700:PRINT",Y":RETURN
957 PRINT"1:GOSUB700:PRINT",Y":RETURN
958 PRINT"1:GOSUB700:PRINT",Y":RETURN
959 PRINT"1:GOSUB700:PRINT",Y":RETURN
960 PRINT"1:GOSUB700:PRINT",Y":RETURN
961 PRINT"1:GOSUB700:PRINT",Y":RETURN
962 PRINT"1:GOSUB700:PRINT",Y":RETURN
963 PRINT"1:GOSUB700:PRINT",Y":RETURN
964 PRINT"1:GOSUB700:PRINT",Y":RETURN
965 PRINT"1:GOSUB700:PRINT",Y":RETURN
966 PRINT"1:GOSUB700:PRINT",Y":RETURN
967 PRINT"1:GOSUB700:PRINT",Y":RETURN
968 PRINT"1:GOSUB700:PRINT",Y":RETURN
969 PRINT"1:GOSUB700:PRINT",Y":RETURN
970 PRINT"1:GOSUB700:PRINT",Y":RETURN
971 PRINT"1:GOSUB700:PRINT",Y":RETURN
972 PRINT"1:GOSUB700:PRINT",Y":RETURN
973 PRINT"1:GOSUB700:PRINT",Y":RETURN
974 PRINT"1:GOSUB700:PRINT",Y":RETURN
975 PRINT"1:GOSUB700:PRINT",Y":RETURN
976 PRINT"1:GOSUB700:PRINT",Y":RETURN
977 PRINT"1:GOSUB700:PRINT",Y":RETURN
978 PRINT"1:GOSUB700:PRINT",Y":RETURN
979 PRINT"1:GOSUB700:PRINT",Y":RETURN
980 PRINT"1:GOSUB700:PRINT",Y":RETURN
981 PRINT"1:GOSUB700:PRINT",Y":RETURN
982 PRINT"1:GOSUB700:PRINT",Y":RETURN
983 PRINT"1:GOSUB700:PRINT",Y":RETURN
984 PRINT"1:GOSUB700:PRINT",Y":RETURN
985 PRINT"1:GOSUB700:PRINT",Y":RETURN
986 PRINT"1:GOSUB700:PRINT",Y":RETURN
987 PRINT"1:GOSUB700:PRINT",Y":RETURN
988 PRINT"1:GOSUB700:PRINT",Y":RETURN
989 PRINT"1:GOSUB700:PRINT",Y":RETURN
990 PRINT"1:GOSUB700:PRINT",Y":RETURN
991 PRINT"1:GOSUB700:PRINT",Y":RETURN
992 PRINT"1:GOSUB700:PRINT",Y":RETURN
993 PRINT"1:GOSUB700:PRINT",Y":RETURN
994 PRINT"1:GOSUB700:PRINT",Y":RETURN
995 PRINT"1:GOSUB700:PRINT",Y":RETURN
996 PRINT"1:GOSUB700:PRINT",Y":RETURN
997 PRINT"1:GOSUB700:PRINT",Y":RETURN
998 PRINT"1:GOSUB700:PRINT",Y":RETURN
999 PRINT"1:GOSUB700:PRINT",Y":RETURN
1000 PRINT"1:GOSUB700:PRINT",Y":RETURN

```

decimal en hexadecimal e imprime los caracteres hexadecimales.

Línea 698: Esta rutina nos da la forma de salirnos de un "loop" For... Next y de volver al menú. Si salimos de esta subrutina sin hacer un "RETURN", o si un "loop" For... Next es acabado demasiado pronto, el registro de "stack" del 6502/6510 se llenará de basura y el resultado es un error de "Out Of Memory" cuando, sin embargo, todavía hay memoria disponible. Esta rutina limpia todos los internos de la máquina cuando salimos del "loop".

Líneas 700-782: Esta rutina contiene la opción de desensamblaje. La línea 700 llama la subrutina de activación de la impresora, y luego inicializa los "loops" For... Next desde el principio hasta el final de las direcciones de memoria. Si la impresora es activada, el programa saltará a la línea 720. En caso contrario, la línea 705 es ejecutada y llamará a una rutina de interrupción en la línea 950 para comprobar una condición de parada de salida.

El programa a continuación cargará el valor desde la memoria y utilizará este valor para apuntar a un elemento del "array" 0%. El elemento del "array" será dividido a continuación por 100 para apuntar dentro del mnemónico dentro del lenguaje "assembler" en el "array" M\$. El resultado se utilizará en la línea 720 para dirigir el programa a una subrutina correcta e imprimir el operando en el modo de direccionamiento requerido del 6502/6510. Estas subrutinas están ubicadas desde la línea 725 a la línea 782.

Líneas 783-798: Estas líneas contienen los valores de datos utilizados para cargar el "array" M\$ y el "array" 0% cuando el programa se ejecuta por primera vez.

Líneas 800-810: Estas líneas contienen la rutina de la opción de corrección (patch). En la línea 800 empieza un "loop" For... Next desde la dirección de principio hasta la dirección de final. A continuación el programa llama la rutina de conversión decimal a hexadecimal en la línea 950 e imprime la posición de memoria. La subrutina de entrada hexadecimal es llamada en la línea 820, y los datos introducidos se escriben en la posición de memoria en la línea 810.

Líneas 820-890: Esta rutina introduce dos caracteres hexadecimales desde el teclado y convierte su valor en un número decimal. La rutina pone a 1 un "flag" de "RETURN" (R) si una salida de esta rutina es solicitada pulsando la tecla "X".

Líneas 895 # 898: Esta rutina es la subrutina de solicitud de interrupción llamada por las opciones de volcado y desensamblaje para comprobar si ha habido una solicitud de parada de salida.

Líneas 900-910: Esta rutina contiene la opción de volcado. La línea 900 llama la subrutina que activa la impresora, y luego inicializa el "loop" For... Next para la dirección de principio y final. El programa luego comprueba la condición de

Listado 1 (continuación)

```

905 IFL=0 THEN GOSUB 895: IFR THEN 696
910 H=PEEK(IP): PRINT " (256) " : GOSUB 920: NEXT: PRINT: RETURN
920 W=INT (H/16): H=H-W*16: GOSUB 930: W=H
930 W=W*16: IF W=57 THEN W=W*7
940 PRINT CHR$(W): RETURN
950 H=INT (IF/256): Z=P-H*256: GOSUB 920: H=Z: GOTO 920
970 PRINT " SI? " : GOSUB 980: IFR THEN RETURN
975 GOTO PRINT " "
980 H$="" : GOSUB 985: IFR THEN RETURN
985 H$=H$+H: E=H: H=H/256: PRINT: RETURN
990 L=0: PRINT " SALIDA (ESC) A (ESC) "
991 INPUT " IMPRESORA (ESC) " : IF L=1 THEN L=0: IF L=0 THEN L=1: OPEN "A", 4, 0: CMD4
992 RETURN

```



final de pantalla, y luego en la línea 950 llama la rutina de conversión de decimal a hexadecimal. La línea 905 llama una subrutina de solicitud de interrupción en la línea 895 si una salida es presentada en la pantalla, y luego se carga un valor desde una posición de memoria y luego se llama la rutina de conversión decimal a hexadecimal y se imprime con la rutina en la línea 920.

Líneas 920-950: Esta rutina convierte un número decimal en hexadecimal. Si se entra en la línea 950, cuatro caracteres hexadecimales se imprimirán. Si se entra en la línea 920, se imprimirán dos caracteres hexadecimales.

Líneas 970-975: Esta rutina es la dirección de principio y de final de la rutina de entrada. El mensaje ¿Si? se imprimirá en la línea 970, y luego el programa llamará en la línea 980 y 985 la rutina de cálculo de conversión de dirección hexadecimal a dirección decimal.

La línea 980 limpia la variable H\$, y llama la rutina de entrada del carácter hexadecimal en la línea 920 y luego ejecuta la línea 985 para convertir el carácter hexadecimal a número decimal.

Líneas 990-993: Esta subrutina imprime el mensaje de salida a impresora y activa la impresora cuando se contesta por un Si desde el teclado.

Conclusión

El programa "Periscopio" es una herramienta útil, un monitor simple para entrar y presentar programas escritos en lenguaje máquina. Su finalidad es la de ayudar a los nuevos usuarios del Commodore 64 y del VIC-20 para conocer mejor su máquina. Al mismo tiempo proporciona una forma para entrar programas cortos escritos en código máquina del 6502/6510 en el equipo. Con este programa funcionando, y con la "Guía de Referencia del Programador" para el Commodore 64 o el VIC-20, las formas de trabajar internas de la máquina serán mucho más comprensibles. ■

Para todos los usuarios de la informática personal

MS

MicroSistemas

250 ptas.

Edición mensual
de Computerworld/España

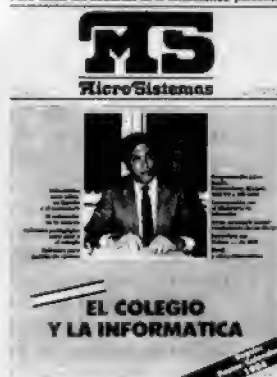
Para todos los usuarios de la informática personal



Ejemplar atrasado
250 pesetas

Colección completa
(6 ejemplares)
1.250 pesetas

Para todos los usuarios de la informática personal



DISK-O-VIC

Se le contagiará la Fiebre del Sábado Noche con este programa de utilidad que proporciona 13 mandatos relacionados a discos y hará bailar el VIC-20 y la unidad de discos 1541. Haz un hueco, John Travolta.

El Commodore VIC-20 y la unidad de discos 1541 constituyen una combinación muy importante. Por supuesto, el VIC-20 es un ordenador muy completo basado en 6502, que ofrece muchas características profesionales como un juego completo de mandatos Basic, un teclado profesional y opciones para una memoria ampliada.

A menudo, un sistema inicial lleva una unidad de cassette como método de almacenamiento en general, pero a medida que asciende el nivel del usuario y aumenta la necesidad de un E/S más rápido, se hace imprescindible una unidad de discos. Igual que las demás unidades de discos flexibles de Commodore, la uni-



dad 1541 es inteligente. Esto significa que por separado la unidad 1541 es un ordenador capaz de realizar muchas funciones sin la intervención del sistema principal.

De hecho, la 1541 contiene su propio microprocesador, un par de VIAs (adaptadores de interface versátiles), 2K de RAM y un sistema de operaciones completo en ROM. Esto con-

duce a dos hechos importantes. Primero, dado que el sistema de la 1541 resulta tan completo, no necesita robar RAM del programa del usuario del sistema principal. Al contrario de muchas combinaciones de unidad de discos/ordenador, el VIC-20 dispone de la misma cantidad de espacio de programa con o sin la unidad de discos.

Segundo, dado que la 1541 es inteligente, puede ser programada externamente para que realice muchas funciones útiles. Realmente, la unidad no tiene límites fijos en el sentido de que si una función determinada no existe de antemano dentro del sistema operativo de discos, se puede escribir un programa para generar tal función.

VIC-20

1514 Disk Drive
Editor/assembler o
machine language

DISK-O-VIC

Este artículo describe un paquete de utilidad que se llama DISK-O-VIC, que añade trece comandos nue-

vos relacionados con discos al VIC-20. Dichos comandos formarán parte del Basic y pueden ser utilizados en la modalidad inmediata para simplificar las operaciones rutinarias con la unidad de discos. Algunos de los comandos, como DLOAD y DSAVE, son ampliaciones de comandos Basic. Otros, como "Screatch" y "Rename", se utilizan para mantener los discos limpios y en orden. Finalmente, otro grupo añade características como listados de mensajes de error, representación del directorio, etc.

El paquete de utilidad DISK-O-VIC está escrito en lenguaje máquina para que sea más rápido y más flexible. Después de cargarse e inicializarse puede quedarse allí durante toda una sesión de programación. Debido a la característica especial de carga (detallada más adelante), DISK-O-VIC se sitúa al principio de la memoria y quedará libre de interferencia del programa Basic. De esta forma, se añaden trece nuevos comandos sin afectar al sistema operativo normal.

La característica especial de carga también hace posible el uso de este paquete en un VIC-20 sea cual sea la cantidad de memoria adicional. No se vuelve anticuado aunque más adelante se añada una memoria adicional. Después de instalarse e inicializarse, la DISK-O-VIC consume 980 bytes y deja intacta la página cero.

Trece Comandos Nuevos en Disco

Antes de entrar en detalles sobre la manera de operación de la DISK-O-VIC, examinaremos los nuevos comandos para saber exactamente lo que hacen. (Para más detalles, vea la tabla de comandos).

Cada vez que un disco flexible se introduce en la 1541 y para luego tener acceso a dicho disco, se crea una tabla especial, llamada el mapa de disponibilidad de bloques, en la memoria de la unidad. Esta tabla contiene información especial sobre el disco que se encuentra en la unidad en un momento dado, como, por ejemplo, la forma en que se ha realizado la partición del disco, los bloques que se encuentran libres y otros asuntos de asignación. Afortunadamente, la unidad de discos se encarga de esta información un tanto esotérica, de forma que vd. se tendrá que preocupar pocas veces por ella.

El proceso de crear esta tabla se llama inicialización. Un disco tiene que inicializarse para que se escriba y se lea de forma correcta. (Tenga en cuenta que unos sistemas de unidad de discos que no son de Commodore utilizan el término inicialización para indicar "dar formato al disco", un proceso que puede escribir encima de

Listado 1. DISK-O-VIC, un programa de utilidad en disco para el VIC-20. Nota: Este programa utiliza el lenguaje ensamblador; se necesita un editor/ensamblador para ejecutarlo.

LINE#	LOC	CODE	LINE
00001	0000		*****
00002	0000		;
00003	0000		;
00004	0000		DISK-O-VIC
00005	0000		DISK UTILITY PACKAGE
00006	0000		FOR THE VIC-20 COMPUTER
00007	0000		;
00008	0000		;
00009	0000		;
00010	0000		;
00011	0000		;
00012	0000		*****
00013	0000		;
00014	0000		;
00015	0000	PARPTR = \$0B	PUNTERO DE ANALIZADOR
00016	0000	POINTR = \$22	PUNTERO DE PROGRAMA DE UTILIDAD DEL VIC-20
00017	0000	BASIC = \$2B	COMIENZO DE PUNTERO DE BASIC
00018	0000	VARIABLE = \$2D	COMIENZO DE PUNTERO DE VARIABLES
00019	0000	CHRGET = \$73	RUTINA CHRGET EN BASIC
00020	0000	CHRGOT = \$79	RE GET CHARACTER ANTERIOR
00021	0000	CHRPTR = \$7A	PUNTERO CHRGET
00022	0000	WEDGE = \$7E	BIFURCACION CHRGET DE BASIC
00023	0000	ST = \$90	LINEA DE ESTADO "ST"
00024	0000	CHRNOS = \$B7	NUMERO DE CARACTERES EN NOMBRE
00025	0000	DEVICE = \$BA	DISPOSITIVO ACTUAL
00026	0000	UTILPT = \$FB	PUNTERO DE PROGRAMA DE UTILIDAD DISK-O-VIC
00027	0000	FIRCHR = \$FD	SALVAR PROVISIONALMENTE PARA CATALOGO
00028	0000	SECCHR = \$FE	SALVAR PROVISIONALMENTE PARA CATALOGO
00029	0000	STACK = \$0100	"STACK" DEL SISTEMA
00030	0000	BUFFER = \$0200	ALMACENAMIENTO INTERMEDIO DE ENTRADA
00031	0000	CMDBUF = \$033C	ALMACENAM INTERM DE COMANDOS EN DISCO
00032	0000	FLAG = CMDBUF	DISTINTIVO PARA CATALOGO
00033	0000	WARMST = \$C474	REARRANQUE DEL BASIC
00034	0000	WAIT = \$C4BC	ESPERAR COMANDO DE SALIDA
00035	0000	INFIN = \$C49F	TERMINAR INTRODUCIR LINEA
00036	0000	CHAIN = \$C533	RECONSTRUIR SERIE DE LINEAS EN BASIC
00037	0000	CLR = \$C659	AJUSTAR PUNTEROS, RESTAURAR BASIC
00038	0000	INTEGR = \$C96B	RECOGER ENTRADA DE ENTEROS
00039	0000	PSTRNG = \$CB1E	IMPRIMIR SERIE SEÑALADA POR (A,Y)
00040	0000	ERROR = \$CF0B	RUTINA DE ERROR DE SINTAXIS
00041	0000	PRLINE = \$DDCD	IMPRIMIR NUMERO DE LINEA
00042	0000	CHROUT = \$E742	REPRESENTAR EN PANTALLA
00043	0000	RESET = \$FD22	ARRANQUE DEL VIC-20
00044	0000	SECLIS = \$FF93	SECUNDARIO DESPUES DE ESCUCHAR
00045	0000	TKSA = \$FF96	SECUNDARIO DESPUES DE HABLAR
00046	0000	ACPTR = \$FFA5	ACEPTAR DATOS DE RS-232
00047	0000	CIOUT = \$FFAB	BYTE DE SALIDA A RS-232
00048	0000	UNTALK = \$FFAB	DESCONECTA EL PERIFERICO
00049	0000	UNLIST = \$FFAE	DESCONECTA EL PERIFERICO
00050	0000	LISTEN = \$FFB1	ORDENAR DISPOSITIVO QUE HABLE
00051	0000	TALK = \$FFB4	ORDENAR DISPOSITIVO QUE HABLE
00052	0000	SETLFS = \$FFBA	FIJAR FICHERO LOGICO
00053	0000	SETNAM = \$FFBD	FIJAR EL NOMBRE DEL FICHERO
00054	0000	OPEN = \$FFC0	ABRIR UN FICHERO
00055	0000	CLOSE = \$FFC3	CERRAR UN FICHERO
00056	0000	CLRCHN = \$FFCC	RESTAURAR DISPOSITIVOS DE FALLOS
00057	0000	INPUT = \$FFCF	INTRODUCIR UN BYTE
00058	0000	OUTPUT = \$FFD2	SALIDA DE BYTE
00059	0000	LOAD = \$FFD5	CARGAR RAM DEL DISPOSITIVO
00060	0000	SAVE = \$FFD8	SALVAR RAM EN DISPOSITIVO
00061	0000	STOP = \$FFE1	COMPROBAR TECLA DE STOP
00062	0000	GETIN = \$FFE4	OBTENER PULSACION
00063	0000		
00064	0000		
00065	0000		* = \$1000
00066	1000		
00067	1000		
00068	1000		
00069	1000		INICIALIZACION DISK-O-VIC
00070	1000		
00071	1000		
00072	1000	A2 00	LDX #SYSTEM1-SYSTEM1 ,INDICE NUEVO SISTEMA
00073	1002	A0 00	LDY #00
00074	1004	B0 54 13	PUTIN LDA SYSTEM1,X ,CARGAR CODIGO OPERATIVO E
00075	1007	F0 07	BEQ INSTAL ,INSTALAR EN BIFURCACION
00076	1009	99 7E 00	STA WEDGE,Y ,CERO SIGNIFICA FIN DE BIFURCACION
00077	100C	EB	INY
00078	100D	EB	INX
00079	100E	D0 F4	BNE PUTIN ,INTRODUCIR SIGUIENTE CODIGO OPERATIVO
00080	1010	4C 74 C4	INSTAL JMP WARMST ,BIFURCAR SIEMPRE
00081	1013		
00082	1013		
00083	1013		
00084	1013		COMIENZO DE INTRODUCCION DE BIFURCACION
00085	1013		
00086	1013	B0 0A	INSERT BCS NORM ,FIN DE CHRGET NORMAL
00087	1015	C9 20	CMF #020
00088	1017	F0 32	BEQ REGET ,IGNORAR ESPACIOS
00089	1019	3B	SEC
00090	101A	E9 30	SBC #030
00091	101C	3B	SEC
00092	101D	E9 D0	SBC #0D0
00093	101F	BD BE 13	NORM STA ASAVE ,SALVAR ACUMULADOR
00094	1022	0B	PHP
00095	1023	6B	PLA

Sigue

datos o destruirlos. Esto no es el caso con la unidad de disco 1541).

Para asegurarse de que la información contenida en la memoria de la unidad esté actualizada, el disco se debe de inicializar con frecuencia durante una sesión. Según sale de la fábrica, la 1541 normalmente realiza el proceso de auto-inicialización durante la ejecución de varios coman-

dos. Con la finalidad de tener un margen de seguridad, un proceso automático de inicialización precede cada mandato en DISK-O-VIC. Aunque esto parezca un poco excesivo, el proceso sólo tarda un segundo y ayuda enormemente a reducir el número de problemas. Nunca hace daño inicializarse de sobra.

Los comandos DLOAD y DSAVE

funcionan exactamente igual que los comandos "Load" y "Save" del VIC-20, sólo que el ordenador sabe automáticamente que el dispositivo correcto para tener acceso es la unidad de disco (dispositivo número ocho). Estos comandos solamente se destinan a programas Basic.

Nunca se deben usar los comandos DLOAD o DSAVE en programas de

Tabla 1. Explicación de los comandos de DISK-O-VIC

"APPEND" (Agregar). Este comando permite que un programa que se encuentra en disco se agregue al final de otro en la memoria. Para que esto no se complique demasiado para el VIC-20, es importante que el programa en la memoria tenga los números de línea menores que los que tiene el programa en disco que se agrega. La disponibilidad de este comando facilita la construcción de grandes bibliotecas de subrutinas a partir de las cuales se pueden construir programas completos. La sintaxis correcta es: APPEND "título de programa" (retorno)

Igual que ocurre con el Basic normal del VIC-20, se pueden abreviar algunas palabras. Por ejemplo, en vez de teclear la palabra completa APPEND, se puede teclear "A shift-P". (Todos los comandos DISK-O-VIC se pueden abreviar de esta forma. Simplemente se teclea la primera letra del comando, seguida por la segunda letra en "shift").

CATALOG (Catalogar). Para saber lo que contiene el disco que se encuentra actualmente en la unidad de disco, se mecanografía CATALOG y se pulsa la tecla de retorno. Un listado de directorio se imprime en la pantalla mostrando todos los programas disponibles. Hay que tener en cuenta que, a diferencia del método empleado normalmente por la 1541 para consultar el directorio, CATALOG no molesta el programa que se encuentra en la memoria del VIC-20.

Se ha agregado una característica especial de hacer una pausa, para que resulte más fácil. Se pulsa la barra de espaciado una vez para hacer una pausa en el listado. Se vuelve a pulsar para continuar. También se puede pulsar la tecla "run/stop" para parar un listado.

COLLECT (Recoger). Si se mecanografía este comando y se pulsa la tecla de retorno, el disco que se encuentra en la unidad será validado o recogido. En términos sencillos, esto hace que la unidad busque por todo el disco, comprobando que todos los bloques se han "conectado" correctamente. Cualquier bloque que ha sido asignado de forma incorrecta será despejado y preparado para servir de almacenamiento. La operación completa de este comando es bastante compleja, pero fundamentalmente lo que hace es revisar y despejar el disco. Igual que la operación de inicialización, nunca hace daño realizar el comando "Collect" con frecuencia.

DLOAD. Este funciona igual que el comando normal de LOAD pero remite el fallo automáticamente a la unidad de discos. Por ejemplo, se mecanografía el "nombre del programa" DLOAD y se pulsa la tecla de retorno. La unidad se inicializa automáticamente, y se comprueban el programa cargado y los errores de disco. Para poner esto en perspectiva podemos decir que DLOAD es equivalente a los siguientes pasos:

OPEN 1, 8, 15, "I"
LOAD "program name", 8
INPUT#1 disk error message, etc.
CLOSE 1

Es evidente que aunque DLOAD es un comando simple, realiza mucho. Además, DLOAD sólo puede ser utilizado en programas Basic o programas de lenguaje de máquina que se "parecen" a Basic. Esta limitación se debe al hecho de que el VIC-20 tiene un extremo formado de carga de "memoria deslizante", (sliding memory)

DSAVE. Esto es igual que DLOAD pero salva el programa Basic en disco. Se realizan las mismas operaciones de inicialización y detección de errores.

HEADER. Esto es un comando especial que da formato a un disco virgen para su uso posterior. Se imprimen marcas magnéticas en el disco que sirven de guía al 1541, y se asignan un título y un código de identificación al disco. La sintaxis para su uso es la siguiente:

HEADER. "nombre del disco", lxx (retorno) donde "nombre del disco" es el nombre asignado al disco. "xx" se utiliza aquí como el código de identificación. Sin embargo, se puede utilizar cualquier combinación de dos caracteres. Tenga en cuenta que la coma es necesaria, igual que la letra "I". Antes de realizar un comando HEADER en el disco, se imprime la pregunta "Estás seguro? (Si/No)" en la pantalla. Una respuesta de "S" (SI) inicia el comando; cualquier otra respuesta cancela anormalmente el proceso. Dado que el comando HEADER escribe encima del disco, es importante proporcionar esta característica de "Estás seguro?".

INIT. Como hemos indicado antes, cada comando de DISK-O-VIC incorpora la operación de inicialización automática. Sin embargo, hay ocasiones en que un disco está causando problemas y resulta necesario forzar un proceso de inicialización. Para hacer esto, se mecanografía INIT, se pulsa la tecla de retorno y el disco será inicializado. Este comando es igual que teclear:
OPEN 1, 8, 15, "I"
CLOSE 1

KILL. Este es un comando de auto-destrucción. Si vd. se ha hartado de utilizar DISK-O-VIC durante una sesión de programación y quiere eliminarlo totalmente del ordenador, teclee KILL y pulsa la tecla de retorno. El ordenador ejecutará un proceso entero de restauración, comportándose como si se hubiera apagado y vuelto a encenderse. No confunda este comando con el de "OFF" (a continuación). KILL restaura el ordenador totalmente. En general, utilice este comando solamente cuando quiere provocar un arranque.

OFF. Este comando apaga DISK-O-VIC, pero sigue dentro de la memoria, a salvo y protegido. Así puede volver a usarse en cualquier momento. Dado que DISK-O-

VIC reduce un poco la velocidad de Basic, se puede apagar cuando se ejecuta un programa para poder alcanzar la máxima velocidad. Para volver a usarlo se teclea:
SYS 256*PEEK(56)+PEEK(55) (retorno)

RENAME. Este comando vuelve a nombrar un programa en disco, sin afectar ningún programa que se encuentre en la memoria. Por ejemplo,
RENAME "nombre antiguo" TO "nombre nuevo" (retorno)

Cambiará el nombre del programa al "nombre nuevo". Hay que tener en cuenta varias cosas. El nombre antiguo viene primero, y luego el nuevo. La palabra "TO" tiene que estar presente entre los dos nombres para que funcione el comando. Finalmente, se proporciona una facilidad de detección de errores para que resulte imposible volver a nombrar un fichero con un nombre que se está utilizando.

SCRATCH. Con este comando se puede borrar (SCRATCH) un fichero o programa del disco. Se mecanografía SCRATCH seguido del nombre del fichero, y se pulsa la tecla de retorno. Una vez más, se presenta la pregunta "Estás seguro? (Y/N)". Se borra el fichero si se recibe la respuesta "Y".

SEND (Enviar). Este es un comando de fines generales y se puede utilizar para enviar los comandos en disco estándar de Commodore a la unidad. Por ejemplo,
SEND "I" (retorno)

enviará la letra I al disco, así provocando una inicialización. (Por supuesto, el comando INIT, de DISK-O-VIC, realiza la misma función). Otro ejemplo,
SEND "R: nombre nuevo = nombre antiguo" (retorno)

hará que se vuelva a nombrar el fichero "nombre antiguo". Dado que otros comandos en DISK-O-VIC sirven para casi todo, el comando SEND no se utiliza mucho. Pero resulta útil tenerlo a mano para las operaciones más avanzadas de programación en disco. Para que se sepa, Send es equivalente a
OPEN 1, 8, 15, "command"
CLOSE 1

STATUS (Estado). Este comando busca la causa de un defecto en la operación de disco. Si se enciende la luz roja de detección de errores en la unidad de discos, se teclea STATUS y se pulsa la tecla de retorno. Se apaga la luz y un mensaje de error se imprime en la pantalla. El mensaje describe el error y donde se encuentra el problema en el disco (en términos de pista y sector). Si no hay ningún problema no se imprime ningún mensaje. Para comprobar este comando, teclee lo siguiente:
OPEN 1, 8, 1, "BASURA" (retorno)

La unidad 1541 empezará a funcionar, y suponiendo que no existe ningún fichero nombrado "BASURA" en el disco, la luz de errores se debe de encender. Teclee STATUS, y el mensaje de error se imprimirá en la pantalla. Vea el manual para la unidad de discos 1541 para consultar una explicación completa de los mensajes de error.

TENEMOS PROGRAMAS MUY INTERESANTES PARA SU COMMODORE 64

por ejemplo...

CONTABILIDAD

Ideado y realizado de acuerdo con el Plan General Contable español.

Con una capacidad de:
300 cuentas y 3000 apuntes
por diskette.

Se incluye MANUAL DE USUARIO con nociones fundamentales de contabilidad y Plan General Contable Español.

GESTION STOCK

Un extraordinario control de almacén.

Hasta 1000 artículos y
1400 apuntes por diskette.

Salidas por pantalla e impresora. Inventarios. Tarifas de precios. Listado de artículos bajo mínimo. Listado total o parcial del fichero. Cierre periódico. Copias de seguridad. Detallado manual.

BASE DE DATOS

Diseñe su propio fichero solo de acuerdo con sus necesidades. Número de registros variable, (ej.: 5000 de 30 caracteres). Salidas por pantalla o impresora. Búsqueda por campos. Listado total o selectivo. Copias de seguridad, etc.

Esto es una simple exposición de algunos de nuestros programas.

*Si no está "su" programa en este anuncio, o si desea
más información, solicítela a:*

EAF microgestion

consejo de ciento, 563-565
telefono 93-231 95 87

barcelona-13
apdo. 24.143

lenguaje de máquina o híbridos, dado que los comandos llegan a unas conclusiones sobre el comienzo de espacio del programa que pueden o no ser ciertas para programas de lenguaje de máquina. En general, todos los comandos de DISK-O-VIC dan por hecho que se trabaja en Basic.

Los comandos DLOAD y DSAVE automáticamente comprueban el canal de errores una vez realizada una operación para ver si todo ha salido bien. Si se detecta un error (Unidad No Preparada, Fichero Existe, Fichero No Encontrado, etc.), el mensaje se imprime en la pantalla y el fichero se cierra.

"Catálogo" constituye un comando interesante. Al contrario de cómo se hacían las cosas antes, se puede imprimir el directorio de discos o catálogo directamente en la pantalla, así conservando cualquier programa en la memoria. Para concluir la representación del listado en la pantalla, la barra de espaciado se pulsa una vez; para volver a iniciar la representación del listado, la barra de espaciado se vuelve a pulsar.

La finalidad del resto de los comandos debe ser evidente. En caso contrario sería útil repasar la Tabla 1 y consultar el manual de la unidad de discos 1541 de vez en cuando. Los usuarios de los ordenadores Commodore más grandes (y más caros) probablemente conocerán muchos de los comandos. A diferencia del VIC-20, ordenadores como el PET y SUPER-PET ya disponen de un juego de comandos un disco que se parece mucho a los de la DISK-O-VIC.

Cómo Funciona el Programa

Ahora que DISK-O-VIC se ha presentado, vamos a ver cómo funciona. Para que sea de más fácil comprensión, el Listado 1 presenta un listado de ensamblador para el programa completo. Dado que los ensambladores empiezan a ser más comunes para el VIC-20, a lo mejor vd. prefiere introducir el código fuente y ensamblar* su propia versión. Pero el listado de ensamblador se presenta aquí dado su valor didáctico, y la mayoría de los usuarios seguramente preferirán introducir el código objeto directamente. Se presenta un volcado hexadecimal de dicho código en el Listado 2.

Probablemente muchos problemas complicados se pueden resolver estudiando los comentarios en este listado. Sin embargo, para que sirva de ayuda en comprenderlo, describiré la estructura básica del programa. Para poder hacerlo, tenemos que estudiar la forma en que Basic recoge y ejecuta un comando.

Cuando actúa el Basic interpretativo, un puntero tiene que buscar los

Listado 1 continuación

00096	1024	BD	BD	13	STA	SSAVE	,SALVAR REGISTRO DE ESTADO
00097	1027	BE	BF	13	STX	XSAVE	,SALVAR REGISTRO X
00098	102A	BC	C0	13	STY	YSAVE	,SALVAR REGISTRO Y
00099	102D	BA			TSX		
00100	102E	BD	01	01	LDA	STACK+1,X	,EXAMINAR DIRECCIONES DESPLAZADAS
00101	1031	C9	BC		CMF	#WAIT	,PARA VER SI LLEGAMOS DE
00102	1033	D6	07		BNE	COMMON	,ESPERAR (ESPERANDO UNA
00103	1035	BD	02	01	LDA	STACK+2,X	,ROUTINA DE COMANDO)
00104	1038	C9	C4		CMF	#WAIT	
00105	103A	F0	12		BEQ	VCDISK	,SI LO HICIMOS, SIGUE
00106	103C	AC	C0	13	LDY	YSAVE	,NO, NO LO HICIMOS
00107	103F	AE	BF	13	LDX	XSAVE	,TIENE QUE SER UNA OPERACION
00108	1042	AD	BD	13	LDA	SSAVE	,NORMAL DE CHRGET, ASI QUE RESTAURAR TODOS
00109	1045	48			PHA		,LOS REGISTROS Y CONTINUAR
00110	1046	AD	BE	13	LDA	ASAVE	
00111	1049	28			PLP		
00112	104A	60			RTS		
00113	104B	4C	73	00	REGET	JMP CHRGET	
00114	104E				:		
00115	104E				:		
00116	104E				:	PARSER ROUTINE	
00117	104E				:		
00118	104E				:		
00119	104E	AD	BD	13	VCDISK	LDA SSAVE	,MIRA EL ANTERIOR REGISTRO DE ESTADO
00120	1051	4A			LSR	A	,ERA LA ENTRADA UNA LINEA ?
00121	1052	90	3A		BCC	NUMBER	,SI BUSCA EN OTRO SITIO
00122	1054	A2	00		LDX	#000	,NO, ERA UN COMANDO
00123	1056	86	08		STX	PARPTR	,INTRODUCIR POR INDICE EN LOS COMANDOS
00124	1058	A4	7A		LDY	CHRPTR	,DIRECCION DE CARACTER EN COMANDO
00125	105A	B9	00	02	CHECK	LDA BUFFER,Y	,OBTENER EL CARACTER DE COMANDO
00126	105D	38			SEC		
00127	105E	FD	5C	13	SBC	KEYWRD,X	,COMPARAR CON TABLA
00128	1061	F0	13		BEQ	DIRECT	,TOTALMENTE DE ACUERDO
00129	1063	C9	80		CMF	#80	
00130	1065	F0	13		BEQ	SHIFT	,COMPARAR LETRAS MEDIANTE "SHIFT"
00131	1067	E6	08		INC	PARPTR	,SEGUIR PISTA DE POSICION
00132	1069	E8			INX		,NO LA PALABRA CLAVE NECESARIA, ASI QUE
00133	106A	BD	5B	13	LDA	KEYWRD-1,X	,DESPLAZAR X A UN PUNTO (PARA APUNTAR A
00134	106D	10	FA		BPL	MOVE	,SIGUIENTE PALABRA DE LA TABLA
00135	106F	BD	5C	13	LDA	KEYWRD,X	
00136	1072	D0	E4		BNE	PARSER	,BUSCAR SIGUIENTE PALABRA CLAVE
00137	1074	F0	C6		BEQ	COMMON	,PALABRAS CLAVE AGOTADAS
00138	1076	E8			DIRECT	INX	,HASTA AHORA SON IGUALES, PERO
00139	1077	C8			INY		,COMPROBAR SIGUIENTE LETRA
00140	1078	D0	E0		BNE	CHECK	,BIFURCAR SIEMPRE
00141	107A	84	7A		SHIFT	STY CHRPTR	,PUNTA CHRPTR EN ULTIMO BYTE
00142	107C	A5	08		LDA	PARPTR	,BUSCAR POSICION DE COMANDO
00143	107E	0A			ASL	A	,MULTIPLICAR POR DOS DADO QUE
00144	107F	AA			TAX		,DIRECCION SE COMPONE DE DOS BYTES
00145	1080	BD	A4	13	LDA	ACTION+1,X	,CARGAR LA DIRECCION CORRECTA
00146	1083	48			PHA		,DE ROUTINA DESPLAZANDOLA
00147	1084	BD	A3	13	LDA	ACTION,X	,EN EL "STACK"
00148	1087	48			PHA		
00149	108B	20	3C	10	JSR	COMMON	,RESTAURAR TODOS LOS REGISTROS
00150	108B	4C	73	00	JMP	CHRGET	,TERMINAR ENTRADA DE LINEAS
00151	108E				:		
00152	108E				:		
00153	108E				:		
00154	108E				:		
00155	108E				:		
00156	108E				:		
00157	108E				:		
00158	108E				:		
00159	108E				:		
00160	108E				:		
00161	108E	68			NUMBER	PLA	,INCAPACITAR DIRECCION NORMAL RTS
00162	108F	68			PLA		
00163	1090	20	3C	10	JSR	COMMON	,RESTAURAR TODOS LOS REGISTROS
00164	1093	20	68	C9	JSR	INTEGER	,OBTENER NUMERO DE LINEA
00165	1096	4C	7F	C4	JMP	INFIN	,TERMINAR ENTRADA DE LINEA
00166	1099				:		
00167	1099				:		
00168	1099				:		
00169	1099				:		
00170	1099				:		
00171	1099	A0	02		APPEND	LDY #02	
00172	109B	A5	2D		DECREM	LDA VARBLE	,REDUCIR DOS VECES EL FINAL
00173	109D	D0	02		BNE	ONCE	,DEL PUNTERO DE PROGRAMA
00174	109F	C6	2E		DEC	VARBLE+1	
00175	10A1	C6	2D		DEC	VARBLE	
00176	10A3	88			DEY		
00177	10A4	D0	FS		BNE	DECREM	,VOLVER A HACERLO SI HACE FALTA
00178	10A6	A5	2D		LDA	VARBLE	,A CONTINUACION SENALA LA ROUTINA DE
00179	10A8	B5	FB		STA	UTILPT	, "APPEND" A ESTA POSICION
00180	10AA	A5	2E		LDA	VARBLE+1	
00181	10AC	B5	FC		STA	UTILPT+1	
00182	10AE	4C	3C	11	JMP	LOADSB	,REALIZAR CAMBIO DE POSICION DE CARGA
00183	10B1				:		
00184	10B1				:		
00185	10B1				:		
00186	10B1				:		
00187	10B1				:		
00188	10B1	20	EA	12	CAT	JSR ICMD	,INICIALIZAR DISCO
00189	10B4	A9	24		LDA	#*	,COMANDO PARA CATALOGO
00190	10B6	8D	3C	03	STA	CHDBUF	
00191	10B9	A2	3C		LDX	#CHDBUF	,OBTENER DIRECCION DE COMANDO
00192	10BB	A0	03		LDY	#CHDBUF	
00193	10BD	A9	01		LDA	#01	,LONGITUD DE COMANDO
00194	10BF	20	BD	FF	JSR	SETNAM	,FIJAR EL NOMBRE
00195	10C2	A9	0E		LDA	#0E	,NUMERO DE FICHERO

Sigue

Listado / continuación

```

00196 10C4 A0 60 LDY #060 .DIRECCION SECUNDARIA
00197 10C6 A2 08 LDX #008 .NUMERO DE DISPOSITIVO
00198 10C8 20 BA FF JSR SETLFS .FIJAR EL FICHERO LOGICO
00199 10CB 20 C0 FF JSR OPEN .ABRIR EL FICHERO
00200 10CE A9 08 LDA #008
00201 10D0 20 B4 FF JSR TALK .INICIAR CONVERSACION
00202 10D3 A9 60 LDA #060 .ENVIAR EL COMANDO AHORA
00203 10D5 20 96 FF JSR TKSA
00204 10D8 A9 00 LDA #000
00205 10DA B5 90 STA ST .SACAR EL ST EN CERO
00206 10DC A0 03 LDY #003
00207 10DE BC 3C 03 THREAD STY FLAG .SALTAR LAS DIRECCIONES INICIALES
00208 10E1 20 A5 FF JSR ACPTL .SALVAR EL CONTADOR DE SALTO
00209 10E4 B5 FD STA FIRCHR .OBTENER UN CARACTER
00210 10E6 A4 90 LDY ST .SALVAR PRIMER CARACTER
00211 10E8 D0 39 BNE BADSTA .COMPROBAR EL ST
00212 10EA 20 A5 FF JSR ACPTL .BIFURCAR SI VAN MAL
00213 10ED B5 FE STA SECCHR .OBTENER SIGUIENTE CARACTER
00214 10EF A4 90 LDY ST .SALVAR SEGUNDO CARACTER
00215 10F1 D0 30 BNE BADSTA .VOLVER A COMPROBAR
00216 10F3 AC 3C 03 LDY FLAG .BIFURCAR SI VA MAL
00217 10F6 B8 DEY .CUANTOS BYTES
00218 10F7 D0 E5 BNE THREAD .HEMOS RECHAZADO?
00219 10F9 A6 FD LDX FIRCHR .NO LOS SUFICIENTES
00220 10FB A5 FE LDA SECCHR .OBTENER PRIMER CARACTER
00221 10FD 20 CD DD JSR PRLINE .OBTENER SEGUNDO CARACTER
00222 1100 20 39 13 JSR SPACE .IMPRIMIR EL NUMERO DE BLOQUES
00223 1103 20 A5 FF JSR ACPTL .IMPRIMIR UN ESPACIO
00224 1106 A6 90 LDX ST .OBTENER CARACTER
00225 1108 D0 19 BNE BADSTA .COMPROBAR EL ST
00226 110A C9 00 CMP #000 .SEGUIR ADELANTE SI SE HA HECHO
00227 110C F0 0E BEQ FINLIN .O-FIN DE LINEA
00228 110E 20 D2 FF JSR OUTPUT .TERMINAR LA LINEA
00229 1111 20 E1 FF JSR STOP .SI NO, IMPRIMIR CARACTER
00230 1114 F0 0D BEQ BADSTA .COMPROBAR LA TECLA STOP
00231 1116 20 27 13 JSR PAUSE .BIFURCAR SI SE EMPUJA
00232 1119 4C 03 11 JSR PAUSE .PAUSAR SI HACE FALTA
00233 111C 20 36 13 JMP CONTEN .IMPRIMIR SIGUIENTE ENTRADA
00234 111F A0 02 JSR CARRET .IMPRIMIR RETORNO DE CARRO
00235 1121 D0 B8 LDY #002 .SEGUIR A LINEA SIGUIENTE
00236 1123 A9 0E BNE THREAD .BIFURCAR SIEMPRE
00237 1125 20 C3 FF LDA #00E .CERRAR FICHERO
00238 112B 20 CC FF JSR CLOSE
00239 112B 4C 74 C4 JSR CLRCN .RESTAURAR DISPOSITIVOS DE FALLOS
00240 112E JMP WARMST .RETORNAR A BASIC
00241 112E ;
00242 112E ;
00243 112E ;*** ENTRADA DE COMANDO "RECOGER" ***
00244 112E ;
00245 112E 20 ED 12 COLLEC JSR CCHD .REALIZAR RECOGIDA
00246 1131 4C 74 C4 JMP WARMST .RETORNAR A BASIC
00247 1134 ;
00248 1134 ;
00249 1134 ;*** ENTRADA DE COMANDO "DLOAD" ***
00250 1134 ;
00251 1134 ;
00252 1134 A5 2B DLOAD LDA BASIC .EL PROGRAMA SE CARGARA
00253 1136 B5 FB STA UTILPT .EN AREA NORMAL
00254 1138 A5 2C LDA BASIC+1 .DE PROGRAMA BASIC
00255 113A B5 FC STA UTILPT+1
00256 113C 20 AE 12 LOADSB JSR PARAMS .OBTENER NOMBRE DE PROGRAMA
00257 113F 20 99 12 JSR SETDIS .FIJAR TODOS PARAMETROS DE DISCO
00258 1142 A9 00 LDA #000 .CERO SIGNIFICA CARGAR
00259 1144 A6 FB LDX UTILPT .CAMBIAR POSICION DE CARGA
00260 1146 A4 FC LDY UTILPT+1 .A ESTE PUNTERO
00261 1148 20 D5 FF JSR LOAD .REALIZAR LA CARGA
00262 114B 20 33 C5 JSR CHAIN .VOLVER A CONSTRUIR LA SERIE
00263 114E A5 22 LDA POINTR .AJUSTAR COMIENZO DE VARIABLES
00264 1150 A6 23 LDX POINTR+1 .PARA QUE APUNTE JUSTO DESPUES DE
00265 1152 18 CLC .LOS TRES BYTES DE CERO AL
00266 1153 69 02 ADC #002 .FINAL DEL PROGRAMA
00267 1155 B5 2D STA VARBLE
00268 1157 90 01 BCC NOADJ
00269 1159 EB INX
00270 115A B6 2E NOADJ STX VARBLE+1
00271 115C 20 59 C6 JSR CLR .DESPEJAR TODOS LOS PUNTEROS, ETC
00272 115F 4C 79 11 JMP CLEAN .IR A DESPEJAR AL FICHERO
00273 1162 ;
00274 1162 ;
00275 1162 ;*** ENTRADA DE MANDATO "DSAVE" ***
00276 1162 ;
00277 1162 ;
00278 1162 20 AE 12 DSAVE JSR PARAMS .OBTENER NOMBRE DE PROGRAMA
00279 1165 20 99 12 JSR SETDIS .FIJAR PARAMETROS DE DISCOS
00280 1168 A5 2B LDA BASIC .SEÑALAR "UTILPT" AL
00281 116A B5 FB STA UTILPT .PRINCIPIO DE PROGRAMA
00282 116C A5 2C LDA BASIC+1
00283 116E B5 FC STA UTILPT+1
00284 1170 A9 FB LDA #UTILPT .PAGINA EN DIRECCION DE "UTILPT"
00285 1172 A6 2D LDX VARBLE .REALIZAR O-X E Y CONTIENEN FIN
00286 1174 A4 2E LDY VARBLE+1 .DE DIRECCION DE PROGRAMA
00287 1176 20 D8 FF JSR SAVE .SALVAR EN DISCO
00288 1179 20 E5 12 JSR SHUT .CERRAR EL FICHERO
00289 117C 20 53 12 JSR DERRR .COMPROBAR ERRORES EN DISCO
00290 117F 4C 74 C4 JMP WARMST .TODO HECHO
00291 1182 ;
00292 1182 ;
00293 1182 ;*** ENTRADA DE COMANDO "HEADER" ***
00294 1182 ;
00295 1182 ;
00296 1182 20 FA 12 HEADER JSR HDISK .PREPARAR COMANDO

```

comandos analizando o barriendo la línea de entrada. El intérprete revisa la línea de entrada, carácter por carácter, en búsqueda de un comando que reconoce. Por lo tanto, para añadir comandos nuevos al Basic hay que colocar una bifurcación en la rutina del análisis, desviando la atención del procedimiento normal del barrido a un procedimiento nuevo. Realmente lo que ocurre es que el analizador está obligado a buscar primero los comandos nuevos. Si no puede comparar un comando con ninguno de la lista nueva, se vuelve a enviar el control al sistema normal donde el comando de entrada se comprueba contra la lista antigua.

El primer bloque de código del Listado 1, las líneas 00072-00080, es la rutina de inicialización. Este código introduce bifurcación en la rutina normal de análisis, y de esta forma el proceso de inicialización sólo tiene que tener lugar al principio de la sesión. Después del proceso de inicialización, el analizador Basic siempre busca primero los comandos.

DISK-O-VIC

El siguiente bloque de código se sitúa en las líneas 00086-00165. Esto es un suplemento del analizador. Como hemos indicado antes, el analizador se dirige a esta rutina cada vez que un comando entra en el ordenador. La instrucción clave en este bloque se encuentra en la línea 00100. El "stack" se analiza por si incluye alguna dirección de "Return" ("RTS"). Si la dirección al principio del "stack" indica que el analizador procede del estado de "esperando un mandato" del VIC-20, si inicia la acción. Si se encuentra alguna otra dirección, el analizador puede continuar con sus actividades normales.

Suponiendo que la prueba se ha concluido con éxito y el VIC-20 realmente espera un comando, la línea de entrada se verifica carácter por carácter. Esto ocurre en el bloque de código etiquetado "Rutina de Análisis", que se encuentra en las líneas 00119-00150. La entrada se comprueba contra la lista de comandos DISK-O-VIC localizada en una tabla en las líneas 00573-00586. Si se encuentran dos iguales, se forma una "dirección de acción", y el control pasa a la subrutina correspondiente.

Subrutinas

La mayor parte del programa se destina a las numerosas subrutinas de comandos. Para que sean más fáciles de encontrar, estas subrutinas se clasifican en orden alfabético, empezando por "Append", luego "Catálogo", etc. Aunque en este momento el programa parezca complejo, la verdad es que resulta bastante fácil de analizar

Listado 1 continuación

00297	1185	20	B0 12	JSR STRING	.AÑADIR NOMBRE DE DISCO
00298	1188	20	73 00	JSR CHRGET	.OBTENER SIGUIENTE CARACTER
00299	118B	C9	2C	CMP #'	.REALIZAR UN "NEED" DE UNA COMA
00300	118D	D0	2D	BNE BAD1	.ERROR DE SINTAXIS
00301	118F	9D	3C 03	STA CHDBUF, X	.INTRODUCIR LA COMA
00302	1192	E8		INX	
00303	1193	20	73 00	JSR CHRGET	.OBTENER IDENTIDAD DE DISCO
00304	1196	C9	49	CMP #'I	.TIENE QUE TENER "I"
00305	1198	D0	22	BNE BAD1	
00306	119A	20	73 00	JSR CHRGET	.OBTENER NUMERO DE IDENTIDAD
00307	119D	F0	1D	BEQ BAD1	.MAL, SI NO SE ENCUENTRA ALLI
00308	119F	9D	3C 03	STA CHDBUF, X	.SALVAR PRIMER DIGITO
00309	11A2	E8		INX	
00310	11A3	20	73 00	JSR CHRGET	.RECORDER SIGUIENTE DIGITO
00311	11A6	F0	14	BEQ BAD1	.MAL, SI NO SE ENCUENTRA ALLI
00312	11AB	9D	3C 03	STA CHDBUF, X	.SALVAR SEGUNDO DIGITO
00313	11AB	E8		INX	
00314	11AC	20	73 00	JSR CHRGET	.CARACTERES EXTRAÑOS?
00315	11AF	D0	0B	BNE BAD1	.SI, MALAS NOTICIAS
00316	11B1	9D	3C 03	STA CHDBUF, X	.CERO SIGNIFICA FIN
00317	11B4	20	0F 13	JSR SURE	.ESTAS SEGURO?
00318	11B7	D0	09	BNE NOGO	.NINGUN CERO SIGNIFICA NO
00319	11B9	4C	32 12	JMP XMIT	.SI, REALIZA HEADER
00320	11BC	4C	0B CF	JMP ERROR	
00321	11BF			BAD1	
00322	11BF			:	
00323	11BF			*** ENTRADA DE COMANDO "INICIALIZAR" ***	
00324	11BF			:	
00325	11BF			:	
00326	11BF	20	EA 12	INIT JSR ICMD	.REALIZAR INICIALIZACION
00327	11C2	4C	74 C4	NOGO JMP WARMST	.TODO HECHO
00328	11C5			:	
00329	11C5			:	
00330	11C5			*** ENTRADA DE COMANDO "KILL" ***	
00331	11C5			:	
00332	11C5			:	
00333	11C5	20	22 FD	KILL JSR RESET	.REALIZAR ARRANQUE
00334	11C8			:	
00335	11C8			:	
00336	11C8			*** "OFF" ENTRADA DE COMANDO "OFF" ***	
00337	11C8			:	
00338	11C8			:	
00339	11C8	A2	04	OFF LDX #SYSTEM2-SYSTEM1	.BORRAR ANALIZADOR ACTUAL
00340	11CA	4C	02 10	JMP CROSS	.COLOCAR CHRGET ANTERIOR EN SU SITIO
00341	11CD			:	
00342	11CD			:	
00343	11CD			*** ENTRADA DE COMANDO "RENAME" ***	
00344	11CD			:	
00345	11CD			:	
00346	11CD	20	FD 12	RENAME JSR RDISK	.PREPARAR COMANDO
00347	11D0	20	79 00	JSR CHRGET	.OBTENER SIGUIENTE CARACTER
00348	11D3	F0	E7	BEQ BAD1	.NO SE DA NINGUN NOMBRE, MALO
00349	11D5	C9	22	CMP #'	
00350	11D7	D0	E3	BNE BAD1	.NINGUNA CITA DE COMIENZO ES MALO
00351	11D9	A5	7A	LDA CHRPT	.SALVAR PUNTERO EN ANTIGUO NOMBRE
00352	11DB	B5	FB	STA UTILPT	
00353	11DD	A5	7B	LDA CHRPT+1	
00354	11DF	B5	FC	STA UTILPT+1	
00355	11E1	20	73 00	TWIST JSR CHRGET	.PASAR POR NUEVO NOMBRE
00356	11E4	F0	DA	BEQ BAD1	.MALO SI NO SE DA NINGUNO
00357	11E6	C9	22	CMP #'	.BUSCAR FINAL DE CITA
00358	11E8	D0	F7	BNE TWIST	
00359	11EA	20	73 00	JSR CHRGET	.A CONTINUACION BUSCAR PALABRA "TO"
00360	11ED	C9	54	CMP #'T	
00361	11EF	D0	CB	BNE BAD1	
00362	11F1	20	73 00	JSR CHRGET	
00363	11F4	C9	4F	CMP #'O	
00364	11F6	D0	C4	BNE BAD1	
00365	11F8	20	73 00	JSR CHRGET	.ENCONTRADO
00366	11FB	20	B0 12	JSR STRING	.COLOCAR PRIMERO EL ULTIMO NOMBRE
00367	11FE	A9	3D	LDA #'=	.INTRODUCIR SENAL DE IGUAL A ()
00368	1200	9D	3C 03	STA CHDBUF, X	
00369	1203	E8		INX	
00370	1204	A5	FB	LDA UTILPT	.RESTAURAR CHRPT ANTERIOR
00371	1206	B5	7A	STA CHRPT	
00372	1208	A5	FC	LDA UTILPT+1	
00373	120A	B5	7B	STA CHRPT+1	
00374	120C	20	B0 12	JSR STRING	.COLOCAR PRIMER NOMBRE EL ULTIMO
00375	120F	A9	00	LDA #00	.BYTE DE CERO PARA EL FINAL
00376	1211	9D	3C 03	STA CHDBUF, X	
00377	1214	4C	32 12	JMP XMIT	.ENVIAR EL COMANDO
00378	1217			:	
00379	1217			:	
00380	1217			*** ENTRADA DE COMANDO "SCRATCH" ***	
00381	1217			:	
00382	1217			:	
00383	1217	20	00 13	SCRATCH JSR SDISK	.FIJAR COMANDO "S"
00384	121A	20	B0 12	JSR STRING	.AÑADIR NOMBRE
00385	121D	A9	00	LDA #00	.COLOCAR BYTE DE CERO AL FINAL
00386	121F	9D	3C 03	STA CHDBUF, X	
00387	1222	20	0F 13	JSR SURE	.ESTAS SEGURO?
00388	1225	D0	29	BNE EGRESS	.NO, SI NO ES CERO
00389	1227	4C	32 12	JMP XMIT	.REALIZAR EL "SCRATCH"
00390	122A			:	
00391	122A			:	
00392	122A			*** ENTRADA DE COMANDO "SEND" ***	
00393	122A			:	
00394	122A			:	
00395	122A	20	AE 12	SEND JSR PARAMS	.OBTENER "STRING" DE COMANDOS

atacando una pequeña función a la vez.

Se presentan unas subrutinas de fines generales hacia el final de programa, en la línea 00452. Normalmente son utilizadas por el resto del programa para recoger los nombres de ficheros, obtener los parámetros de los discos, imprimir mensajes en la pantalla, etc. En general, se les han asignado etiquetas o nombres relacionados a las funciones que realizan.

DISK-O-VIC termina presentando varias tablas de datos y direcciones. Primero se presenta una tabla de palabras clave, como hemos indicado antes. A continuación se presenta una tabla que contiene las direcciones de las subrutinas de comandos. Al final, se crea un juego de variables con la finalidad de salvar registros, etc. Al asignar variables a esta zona, se evita el uso de las importantes posiciones de página cero.

Dado que los detalles a veces confunden el asunto, a continuación se presenta un resumen de la estructura general que acabamos de describir:

—Proceso de inicialización.

—La bifurcación se introduce en el analizador.

—La nueva rutina de análisis.

—Las subrutinas de comandos.

—Las subrutinas de fines generales.

—Los datos, direcciones y variables.

Antes de dejar los aspectos teóricos de DISK-O-VIC, debemos mirar la tabla de ecuaciones en el listado de ensamblador. La tabla utiliza alrededor de una docena de posiciones de página cero, pero dado que se utilizan para sus fines normales, el sistema operativo les hace poco caso. La verdad es que las posiciones \$FB a \$FE son de página cero y el VIC-20 no las utiliza en absoluto.

Dado que se trata de un sistema operativo en disco, se supone que la unidad de cassettes no se utiliza. Esto libera un gran bloque de espacio, empezando por \$033C, que normalmente se emplea como almacenamiento intermedio de cassette. Por lo tanto este bloque se puede utilizar como almacenamiento intermedio para los comandos en disco. Incluso si una unidad de cassettes se une al VIC-20 junto con un disco flexible, no surgirá ningún conflicto dado que pocas veces se tendría acceso a las dos unidades a la vez.

Rutinas ROM

La siguiente en las tablas de ecuaciones constituye un grupo grande de subrutinas incluidas en el conjunto ROM del sistema operativo VIC-20. Aunque vd. no tenga un uso inmediato para el DISK-O-VIC, esta tabla le será útil. El uso de una de estas rutinas ROM estandard en programas

VIC-20

Microprocesador: 6502 de MOS TECHNOLOGY de 8 bits.

Memoria: 5 Kbytes de RAM ampliables a 32 K 20 Kbytes de ROM ampliables a 28 K.

Pantalla: 23 líneas de 22 caracteres. Modulador para conectar a un televisor normal. Salida monitor video. **Colores:** 8 para el marco, 16 para el fondo de la pantalla y 8 para los caracteres individuales, video inverso. **Gráficos:** Semi-gráficos por teclado y alta resolución por redefinición del generador de caracteres (situado en RAM). Definición de 176 por 184 puntos. **Teclado:** Tipo QWERTY de 62 teclas más cuatro de función definibles por el usuario.

Sonido: Tres voces de tres octavas cada una decaladas una octava entre sí, resultando una extensión total de cinco octavas. Un generador de ruido aleatorio afinable para efectos especiales, un control general de volumen.

Programación: Lenguaje BASIC, intérprete residente en ROM de 8K. Posibilidad de interceptar las funciones del Basic para crear nuevas instrucciones "a medida". El Basic del Vic es uno de los rápidos actualmente en el mercado.

Complementos: Port de usuario de 8 bits entrada/salida más dos señales de sincronismo.

Bus de expansión para ampliaciones de memoria y periféricos.

Port de juegos con conexión para dos potenciómetros (paddles), y una palanca de juegos (joystick).

Almacenamiento de masa: Unidad de cassette C2N de diseño especial para registrar programas y datos.

Ampliación de memoria: En caso de ser necesario conectar más de un cartucho al mismo tiempo, está disponible un módulo (VIC 1020) que permite la conexión simultánea de hasta seis cartuchos.

VIC-1541 UNIDAD DE DISCO

Capacidad total: 174848 bytes por disco.

Secuencial: 168656 bytes por disco.

Entradas de directorio: 144 por disco.

Sectores por pista: De 17 a 21.

Bytes por sector: 256.

Pistas: 35.

Bloques: 683 (644 bloques libres).

Soportes de información: Discos standar de 5 1/4 pulgadas, de una sola cara y densidad simple.

Sistema operativo: DOS de COMMODORE inteligente (tiene procesador propio y no ocupa memoria del ordenador central).

VIC-1525 IMPRESORA

Método de impresión: Matriz de 5 x 7 puntos, impacto por un solo martillo.

Modo caracteres: Mayúsculas y minúsculas, símbolos, números y caracteres gráficos del VIC-20.

Modo gráfico: Puntos direccionables (bit image). Siete puntos verticales por columna. 480 columna máximo.

Velocidad: 30 caracteres/segundo, de izquierda a derecha, unidireccional.

Caracteres/Línea: Máximo 80. (Posibilidad de impresión en doble ancho).

Espaciado entre líneas: 6 líneas/pulgada —modo caracteres, 9 líneas/pulgada— modo gráfico.

Alimentación de papel: Arrastre por tractor.

Ancho de papel: Entre 4,5 y 10 pulgadas.

Copias: Original más dos copias.

CARTUCHOS

Ayuda programador: Facilita la edición y depuración de programas en Basic. Instrucciones y comandos:

RENUMBER, MERGE, FIND, CHANGE, DELETE, AUTO, TRACE, STEP, OFF, KEY, EDIT, PROG, DUMP, HELP y KILL.

Super expander: Intercepta el Basic del VIC permitiendo incrementar sus instrucciones y comandos en aplicaciones gráficas de sonido y juegos. Instrucciones y comandos: KEY, GRAPHIC, COLOR, POINT, REGION, DRAW, CIRCLE, PAINT, CHAR, SCNCIR, SOUND, RGE, RCOLR, RIXOT, RPOT, RPEN, RJOY y RSND.

Monitor de lenguaje máquina: Facilita enormemente la depuración de programas en lenguaje máquina, es ideal como complemento del Basic para redactar y poner en marcha rutinas de alta velocidad y manejo de datos en tiempo real. Instrucciones y comandos: ASSEMBLE, BREAKPOINT, DISASSEMBLE, ENABLE, VIRTUAL ZERO PAGE, FILL MEMORY, GO, HUNT, INTERPRET, JUMP TO SUBROUTINE, LOAD, MEMORY, NUMBER, QUICK TRACE, REGISTERS, REMOVE BREAKPOINTS, SAVE, TRANSFER, WALK y EXIT TO BASIC. Además existen cartuchos de ampliación de memoria de 3,8 y 16 Kbytes.

CURSO DE INTRODUCCION AL BASIC PARTE I y II.

En forma de libro se ha editado la primera y segunda parte de un curso de Basic que parte "de cero" y está basado en el VIC 20. Van acompañados de dos cassettes con programas y ejercicios para autocontrol.

PLOTTER VIC 1520

Método de impresión: Dibujo mediante bolígrafos de diseño especial.

Color: 4 colores: negro, azul, verde y rojo con cambio desde programa.

Cabezal: Plotter X-Y tipo tambor.

Velocidad de impresión: Media de 14 car./seg.

Caracteres por línea: Máximo 80 carac., formatos de 80, 40, 20 y 10 carac./línea.

Juego de caracteres: 96.

Velocidad de dibujo: 264 pasos/seg.

Longitud del paso: 0,2 mm. en dirección X e Y.

Velocidad de dibujo de línea: 52,8 mm./seg. en dirección X e Y, 73 mm./seg. en una línea a 45 grados.

Area de dibujo: 480 pasos (96 mm.) en dirección X. Programable en dirección Y (Máx. + — 999 de una sola vez).

Papel: Rollo de 4,5 pulgadas (114 mm.).

MONITOR EN COLOR C-1701

Pantalla: 13 pulgadas (330 mm.).

Capacidad de representación: 25 líneas de 40 caracteres.

Resolución: 320 líneas horizontales.

Compatibilidad: VIC 20 y COMMODORE 64.

Conectable a un registrador de video.

Amplificador y altavoz: Incorporados.

commodore
COMPUTER



microelectrónica
y control, s.a.

PEC

Taquigrafo Serra, 7 5.º Telf. 250 51 03. BARCELONA-29
Princesa, 47 3.º G. Telf. 248 95 70. MADRID-8

fabricados por uno mismo, permitirá el ahorro de centenares, hasta miles, de bytes. Vamos a examinar unas cuantas rutinas con más detalles.

Las rutinas ROM del VIC-20 pueden clasificarse en dos categorías generales. La primera, llamada Kernal, incluye las rutinas principales para operaciones de entrada y salida. (Aunque no se sepa por qué, la forma de escribir "Kernal" es la que se acepta oficialmente en Commodore). Estas rutinas Kernal se consideran especiales porque varios modelos de ordenadores Commodore incluyen las mismas rutinas que se encuentran en las mismas direcciones.

Por ejemplo, la rutina de "salida de byte" tiene lugar en \$FFD2 en todos los modelos de ordenadores Commodore. Sin embargo, en general las rutinas Kernal son idénticas solamente en el VIC-20 y el Commodore 64. Esto significa que el software que emplea mucho las rutinas Kernal debe ser de fácil transferencia del VIC-20 al Commodore 64, y al revés.

Los comandos Basic forman otra categoría de rutinas. Las posiciones de éstos serán diferentes en cada máquina, pero parecidos para los ordenadores Commodore. Un ejemplo de estas rutinas es la que se encuentra en \$CBIE, etiquetada PSTRNG en el listado 1. Cuando se solicita, se imprime un "string" en pantalla señalado por el acumulador y el registro "y". Sigue imprimiendo la serie de caracteres hasta que se detecta un cero final.

La tabla de ecuaciones del listado 1 proporciona información adicional sobre el funcionamiento del VIC-20. En la mayoría de los casos, las posiciones y las rutinas llevan etiquetas para facilitar la interpretación, y los comentarios proporcionan más detalles. Si desea saber más sobre las rutinas Kernal, vea "La Guía de Referencia para el Programador del VIC-20" (Howard W. Sams and Co., PO Box 7092, Indianapolis, IN 46206). Para poder comprender las rutinas no Kernal, hace falta un mapa de memoria más extensa.

Introducir el Programa

Ya que hemos hablado de la teoría y la operación del DISK-O-VIC, tenemos que considerar el lado práctico de las cosas. El usuario debe de crear una copia en disco del código objeto para que tenga el programa utilidad siempre a mano. Con esta finalidad, el volcado hexadecimal del Listado 2 corresponde al código fuente del Listado 1. Para usarlo, se introducen los números hexadecimales en el VIC-20 y se salva en disco. Por lo tanto, para llamar DISK-O-VIC, se carga y se inicializa el código.

Listado 1 continuación

00396	122D	A9 00	LDA #000	.BYTE DE CERO SIGNIFICA
00397	122F	9D 3C 03	STA CHDBUF, X	.FIN DE "STRING"
00398	1232	20 EA 12	JSR ICMD	.INICIALIZAR DISCO
00399	1235	20 DB 12	JSR RESPON	.FIJAR CANAL DE COMANDOS
00400	1238	A0 00	LDY #000	
00401	123A	B9 3C 03	SHOOT LDA CHDBUF, Y	.OBTENER "STRING" DE COMANDOS
00402	123D	F0 06	BEQ SWEEP	.CERO SIGNIFICA FINAL
00403	123F	20 AB FF	JSR CIOUT	.SI NO, ENVIAR CARACTER
00404	1242	CB	INY	
00405	1243	90 F5	BCC SHOOT	.BIFURCAR SI NO EXISTE ERROR
00406	1245	20 AE FF	JSR UNLIST	.DESCONECTAR EL DISCO
00407	1248	20 53 12	JSR DERROR	.COMPROBAR ESTADO DE DISCO
00408	124B	F0 03	BEQ EGRESS	.BIFURCAR SIEMPRE
00409	124D		;	
00410	124D		;	
00411	124D		;***.ENTRADA DE COMANDO "STATUS"***	
00412	124D		;	
00413	124D		;	
00414	124D	20 53 12	STATUS JSR DERROR	.COMPROBAR ESTADO DISCO
00415	1250	4C 74 C4	EGRESS JMP WARMST	
00416	1253	A9 08	DERROR LDA #00B	.PREPARADO PARA LEER
00417	1255	B5 BA	STA DEVICE	.EL CANAL DE ERRORES
00418	1257	20 B4 FF	JSR TALK	.OBLIGAR EL BUS QUE HABLE
00419	125A	A9 6F	LDA #06F	.ESTO ES CANAL DE ERRORES
00420	125C	20 96 FF	JSR TKSA	.DIRECCION SECUNDARIA DESPUES DE HABLAR
00421	125F	A0 00	LDY #000	
00422	1261	20 A5 FF	XFER JSR ACPTR	.OBTENER BYTE DE BUS
00423	1264	99 3C 03	STA CHDBUF, Y	.SALVARLO AQUI
00424	1267	CB	INY	
00425	1268	C9 0D	CMP #0D	.BUSCAR RETORNO DEL CARRO
00426	126A	D0 F5	BNE XFER	.SI NO, OBTENER SIGUIENTE CARRO
00427	126C	A9 00	LDA #000	.COLOCAR BYTE DE CERO PARA EL FINAL
00428	126E	99 3C 03	STA CHDBUF, Y	
00429	1271	20 AB FF	JSR UNTALK	.DESCONECTAR CANAL
00430	1274	A0 00	LDY #000	.BUSCAR "00" DE ASCII
00431	1276	A9 30	LDA #'0	
00432	1278	D9 3C 03	CMP CHDBUF, Y	
00433	127B	D0 06	BNE DEFER	.SIN DUDA ES UN ERROR
00434	127D	CB	INY	
00435	127E	D9 3C 03	CMP CHDBUF, Y	
00436	1281	F0 15	BEQ NOERR	.NO SE ENCUENTRA NINGUN ERROR
00437	1283	20 E5 12	JSR SHUT	.CERRAR EL FICHERO
00438	1286	20 CC FF	JSR CLRCHN	.RESTAURAR DISPOSITIVOS DE FALLOS
00439	1289	20 36 13	JSR CARRET	.IMPRIMIR RETORNO DEL CARRO
00440	128C	A9 3C	LDA #CHDBUF	
00441	128E	A0 03	LDY #CHDBUF	
00442	1290	20 1E CB	JSR PSTRNG	.IMPRIMIR EL MENSAJE DE ERROR
00443	1293	68	PLA	.NO RETORNAR SI ES MALO
00444	1294	68	PLA	
00445	1295	4C 74 C4	JMP WARMST	.IR A BASIC
00446	1298	60	NOERR RTS	.RETORNAR A RUTINA DE LLAMADA
00447	1299		;	
00448	1299		;	
00449	1299		; SUBROUTINAS DE FINES GENERALES	
00450	1299		;	
00451	1299		;	
00452	1299	20 EA 12	SETDIS JSR ICMD	.INICIALIZAR DISCO
00453	129C	BA	TXA	.ESTO ES LA LONGITUD DE NOMBRE
00454	129D	A2 3C	LDX #CHDBUF	.X E Y CONTIENEN
00455	129F	A0 03	LDY #CHDBUF	.LA DIRECCION DEL NOMBRE
00456	12A1	20 BD FF	JSR SETNAM	.FIJAR NOMBRE DE PROGRAMA
00457	12A4	A9 08	LDA #00B	.NUMERO DE FICHERO LOGICO
00458	12A6	AA	TAX	.NUMERO DE DISPOSITIVO DE DISCO
00459	12A7	A0 00	LDY #000	.SECUNDARIO PARA VOLVER A UBICAR
00460	12A9	B4 90	STY ST	.SACAR EN CERO "ST"
00461	12AB	4C BA FF	JMP SETLFS	.FIJAR PARAMETROS DE FICHERO
00462	12AE		;	
00463	12AE		;	
00464	12AE	A2 00	PARAMS LDX #000	
00465	12B0	B6 B7	STRING STX CHRNOS	.SALVAR LONGITUD ORIGINAL
00466	12B2	20 79 00	JSR CHRGET	.OBTENER NOMBRE DE FICHERO
00467	12B5	F0 1F	BEQ BAD2	.NO SE DA NINGUN NOMBRE
00468	12B7	C9 22	CMP #' "	.SE NECESITA UNA CITA
00469	12B9	D0 1B	BNE BAD2	
00470	12BB	E6 7A	PICKUP INC CHRPTTR	.ACTUALIZAR PUNTERO CHRGET
00471	12BD	D0 02	BNE NOZER	
00472	12BF	E6 7B	INC CHRPTTR+1	
00473	12C1	A0 00	NOZER LDY #000	
00474	12C3	B1 7A	LDA (CHRPTTR), Y	.OBTENER SIGUIENTE CARACTER
00475	12C5	F0 0A	BEQ DONE	.CERO SIGNIFICA EL FINAL DEL ALMAC INTERM
00476	12C7	C9 22	CMP #' "	.CITA FINAL AHORA?
00477	12C9	F0 06	BEQ DONE	.SI, "STRING" OBTENIDO
00478	12CB	9D 3C 03	STA CHDBUF, X	.ALMACENAR AQUI PROVISIONALMENTE
00479	12CE	EB	INX	
00480	12CF	D0 EA	BNE PICKUP	.BIFURCAR SIEMPRE
00481	12D1	E4 B7	CPX CHRNOS	.NO PERMITIR NOMBRES DE
00482	12D3	F0 01	BEQ BAD2	.LONGITUD CERO
00483	12D5	60	RTS	
00484	12D6	68	BAD2 PLA	.QUITAR ANTIGUA DIRECCION DE RETORNO
00485	12D7	68	PLA	
00486	12D8	4C 0B CF	JMP ERROR	.IR A RUTINA DE ERRORES
00487	12DB		;	
00488	12DB		;	
00489	12DB	A9 08	RESPON LDA #00B	.OBTENER NUMERO DE DISCO
00490	12DD	20 B1 FF	JSR LISTEN	.HACER QUE EL DISCO ESCUCHE
00491	12E0	A9 6F	LDA #06F	
00492	12E2	4C 93 FF	JMP SECLIS	.SECUNDARIO DESPUES DE ESCUCHAR
00493	12E5		;	

Segue

Listado 1 continuación

00494	12E5					
00495	12E5	A9 08	SHUT	LDA #08	NUMERO DEL FICHERO LOGICO	
00496	12E7	4C C3 FF		JMP CLOSE	CERRAR EL FICHERO.	
00497	12EA					
00498	12EA					
00499	12EA	A9 49	ICMD	LDA #'I	COMANDO DE INICIALIZACION	
00500	12EC	2C		.BYTE #2C		
00501	12ED	A9 56	CCMD	LDA #'V	COMANDO DE VALIDEZ	
00502	12EF	4B		PHA	SALVAR EL COMANDO	
00503	12F0	20 DB 12		JSR RESPON	FIJAR CANAL DE COMANDOS	
00504	12F3	6B		PLA	OBTENER COMANDO Y	
00505	12F4	20 AB FF		JSR CROUT	ENVIARLO	
00506	12F7	4C AE FF		JMP UNLIST	PROVOCAR UN UN-LISTEN	
00507	12FA					
00508	12FA					
00509	12FA	A9 4E	HDISK	LDA #'N	PREPARAR COMANDO "HEADER"	
00510	12FC	2C		.BYTE #2C		
00511	12FD	A9 52	RDISK	LDA #'R	PREPARAR COMANDO "RENAME"	
00512	12FF	2C		.BYTE #2C		
00513	1300	A9 53	SDISK	LDA #'S	PREPARAR COMANDO "SCRATCH"	
00514	1302	A2 00		LDX #000		
00515	1304	9D 3C 03		STA CMDBUF, X	ALMACENAR COMANDO AQUI	
00516	1307	EB		INX		
00517	1308	A9 3A		LDA #' :	DOS PUNTOS COMUNES PARA TODOS	
00518	130A	9D 3C 03		STA CMDBUF, X	ALMACENARLOS AQUI	
00519	130D	EB		INX	PREPARAR PARA SIGUIENTE BYTE	
00520	130E	60		RTS		
00521	130F					
00522	130F					
00523	130F	A2 00	SURE	LDX #000		
00524	1311	BD 3E 13	SPEAK	LDA MESSAG, X	ENVIAR MENSAJE	
00525	1314	F0 06		BEQ SILENT	ESTAS SEGURO?	
00526	1316	20 42 E7		JSR CHROUT		
00527	1319	EB		INX		
00528	131A	D0 F5		BNE SPEAK	BIFURCAR SIEMPRE	
00529	131C	20 CF FF	SILENT	JSR INPUT	BUSCAR S=SI	
00530	131F	C9 53		CHP #'S		
00531	1321	0B		PHP		
00532	1322	20 36 13		JSR CARRET	IMPRIMIR RETORNO DE CARRO	
00533	1325	2B		PLP		
00534	1326	60		RTS		
00535	1327					
00536	1327					
00537	1327	20 E4 FF	PAUSE	JSR GETIN	OBTENER UN BYTE	
00538	132A	F0 09		BEQ NOWAIT	NO HAY. SIGUE	
00539	132C	C9 20		CHP #20		
00540	132E	D0 05		BNE NOWAIT	NINGUN ESPACIO DISPONIBLE	
00541	1330	20 E4 FF	KEYUP	JSR GETIN	AHORA ESPERAR	
00542	1333	F0 FB		BEQ KEYUP	OTRA PULSACION	
00543	1335	60		NOWAIT	RTS	
00544	1336					
00545	1336					
00546	1336	A9 0D	CARRET	LDA #0D	IMPRIMIR RETORNO DE CARRO	
00547	1338	2C		.BYTE #2C		
00548	1339	A9 20	SPACE	LDA #20	IMPRIMIR UN ESPACIO	
00549	133B	4C D2 FF		JMP OUTPUT		
00550	133E					
00551	133E					
00552	133E					
00553	133E					
00554	133E					
00555	133E	0D	MESSAG	.BYTE #0D		
00556	133F	45 53		.BYTE 'ESTAS SEGURO? (S/N)'		
00557	1353	00		.BYTE #00		
00558	1354	4C	SYSTM1	.BYTE #4C	NEW CHRGET.	
00559	1355	13 10		.WORD INSERT		
00560	1357	00		.BYTE #00		
00561	1358	80	SYSTM2	.BYTE #80, #0A	OLD CHRGET.	
00562	1359	0A				
00563	135A	C9		.BYTE #C9, #00		
00564	135B	00				
00565	135C					
00566	135C					
00567	135C					
00568	135C					
00569	135C					
00570	135C					
00571	135C					
00572	135C					
00573	135C	41 50	KEYWRD	.BYTE 'APPEN', #C4		
00574	1361	C4				
00575	1362	43 41		.BYTE 'CATAL', #C7		
00576	1368	C7				
00577	1369	43 4F		.BYTE 'COLLEC', #D4		
00578	136F	D4				
00579	1370	44 4C		.BYTE 'BLDA', #C4		
00580	1374	C4				
00581	1375	44 53		.BYTE 'DSAV', #C5		
00582	1379	C5				
00583	137A	4B 45		.BYTE 'HEADE', #D2		
00584	137F	D2				
00585	1380	49 4E 49		.BYTE 'INI', #D4		
00586	1383	D4				
00587	1384	4B 49 4C		.BYTE 'KIL', #CC		
00588	1387	CC				
00589	1388	4F 46		.BYTE 'OF', #C6		
00590	138A	C6				
00591	138B	52 45		.BYTE 'RENAM', #C5		

Dado que el programa está escrito en lenguaje máquina, hace falta un monitor de lenguaje máquina para introducirlo. El VIC-20 no dispone de un monitor residente, pero se ven cada vez más monitores de acoplamiento. Dos buenas opciones son el VICmon o el Tinymon.

El VICmon, fabricado por Commodore, es el monitor de lenguaje máquina oficial para el VIC-20 y ofrece muchos comandos. Se hace en forma de cartucho (cartridge) (ROM) y se encaja fácilmente en la puesta para ampliación. Por otro lado, el Tinymon, es un monitor de cinta o cargado en disco. La ventaja del Tinymon es que puede ser teclado por el usuario y así se ahorra bastante dinero. No soporta tantos comandos como el VICmon pero eso no importa para el propósito que tenemos aquí. Sólo nos interesan los comandos de S (salvar) y de M (volcado de memoria). Por lo tanto, cualquier monitor sirve nuestros propósitos.

Para hacer una copia de DISK-O-VIC para el ordenador se siguen las instrucciones que se presentan a continuación:

—Desconectar cualquier acoplamiento adicional a la memoria. DISK-O-VIC tiene que introducirse en una máquina "stock".

—Cargar un monitor de lenguaje máquina. Sirve igual un monitor basado en cinta/disco o de "cartridge".

—Se introduce el código objeto mediante perforación utilizando como guía el Listado 2. Se inicia la introducción en la posición \$1000 y se continúa hacia arriba.

—Una vez introducido el código, se modifican las siguientes posiciones. Introducir el byte de datos \$2F en las posiciones \$2D, \$2F y \$31. Se introduce el byte de datos \$15 en las posiciones \$2E, \$30 y \$32. Todas estas posiciones son de página cero.

—El monitor sale en Basic usando el comando X.

—Se salva el programa utilizando el comando normal de "SAVE" del VIC-20. El programa puede salvarse tanto en cinta como en disco.

—Si así se desea, cualquier acoplamiento adicional de memoria puede volver a conectarse.

Así se dispone de una versión completa de DISK-O-VIC lista para usarse. El código que se acaba de introducir y salvar es muy especial. Se puede cargar y ejecutar como cualquier otro programa de Basic. Cuando se ejecuta el programa, un cargador especial automáticamente vuelve a ubicar el DIS-O-VIC al principio de la memoria, sea cual sea la localización. Además, el cargador compensa de forma instantánea cualquier memoria adicional que se encuentre conectada al VIC-20.

VIDEO CASINO

Batalla Naval

**Estrategia de Vida o Muerte
en Alta Mar.**



Si te gusta la estrategia naval o simplemente quieres mejorar tu puntería, intenta navegar por los mares de este juego.

El juego "Batalla Naval", para un jugador en el VIC-20 no ampliado, se basa en el popular juego de "Barquitos". El jugador tiene que hundir los barcos que pertenecen al ordenador antes de que éste hunda los del jugador.

El jugador y el ordenador tienen al principio del juego, cinco barcos cada uno, colocados sobre un tablero. A continuación, se presentan los nombres de los barcos y el número de veces que hay que alcanzarlos para que se hundan:

—Portaaviones	5 veces
—Acorazado	4 veces
—Crucero	3 veces
—Submarino	3 veces
—Destructor	2 veces

Al iniciar el juego el jugador tendrá que elegir entre unos niveles de dificultad del uno al tres. En el nivel uno, el jugador y el ordenador disparan una vez cada uno. En el nivel dos, el ordenador dispara dos veces y el jugador una. En el nivel tres, es casi imposible que gane el jugador, ya que

el ordenador dispara tres veces y el jugador una. La única oportunidad que tiene el jugador de ganar es tener suerte y poder encontrar los cinco barcos enemigos en seguida. El ordenador dispara al azar, pero nunca da dos veces en el mismo sitio.

Después de elegir el nivel de dificultad, el jugador coloca sus barcos en el tablero. El ordenador le pide al jugador que le dé una coordenada inicial para el Portaaviones. El jugador introduce una coordenada con el siguiente formato: una letra de A a J seguida por un número de 0 a 9 (por ejemplo, D7 o J0). A continuación, el ordenador le pide al jugador que entre una H o una V para indicar si el barco se tiene que colocar en posición horizontal o vertical.

Si el jugador da una coordenada no correcta, o intenta colocar un barco encima de otro, o trata de situar un barco de tal forma que se sale del tablero, el programa le dirá "Mala Coordenada" y volverá a empezar. Las coordenadas del barco se dibujan

VIC 20

COMMODORE 64

Después que el jugador haya colocado sus barcos, el ordenador situará los suyos. El jugador no los puede ver, dado que se representan con caracteres programables que tienen el mismo aspecto que los que forman el tablero.

Cuando el ordenador termina de situar sus barcos, le pide al jugador que le indique las coordenadas del primer disparo. A continuación, dispara el ordenador. Tanto los disparos acertados como los fallados se representan en el tablero. Si se hunden primero todos los barcos del jugador, el ordenador dice "¡Gano Yo!" y le enseña la posición de sus barcos al jugador. Cuando el jugador alcanza o

Líneas	
50	Bajar parte superior de memoria y realizar procesos aleatorios
60-80	Establecer caracteres programables
90	Fijar colores: indicar al ordenador que busque la información de caracteres en la RAM
100-120	Fijar nivel de dificultad
130	Inicializar variables
140-200	Dibujar representación en pantalla
210-360	Colocar barcos amigos
370-440	Colocar barcos enemigos
450-580	Jugador dispara a los barcos del ordenador
590-670	Dispara ordenador
680-700	Representar mensajes de barco alcanzado y barco hundido
710-750	Fin de rutina de juego
760	Borrar parte superior de pantalla
770	Mensaje de mala coordenada
780	Cronometrar representación de mensaje y luego borrar
790	Rutina de sonido para los disparos
800	Borrar el tablero del enemigo
810-820	Rutina de sonido para blanco alcanzado
830	Representar blancos alcanzados
840	Nombres de barcos y disparos que faltan para hundir barco.

```

58 POKES5,28:POKES5,28:CLR:POKE143,VAL(13D011W,5,21)
59 FORI=176707679:POKE1,PEEK1+1256000:NEXT
60 FORI=1768707174:POKE1,PEEK1+1256000:1,1:NEXT:POKE17125,255
61 FORI=7632766647:FORJ=707670000:POKE1+J,PEEK(7168+J):NEXT:NEXT
62 PRINTCHR$(1471CHR$(15):POKES687,11):POKES6869,255
100 PRINT:PRINTTAB(71"ELIGED1":PRINTTAB(1SPC13NIVE1SPC10E1SPC10IFCULTAD"
110 PRINT:PRINT"0SPC1ACIL1SPC11-11SPC1DIFC1L12SPC1":INPUTU
120 IFU=100U:STHEN99
130 U=38728151+38678:52=38677:V=38678:P=1
140 PRINTCHR$(1471:FORI=170767000:POKE1,NEXT
150 PRINT:1SPC181254578V1SPC18123456789"
160 FORI=7999708119:POKE1,NEXT
170 FORI=79907080857E2P21POKE1,P2POKE2+11,P=P+1:NEXT
180 FORI=17018:PRINT:NEXT
190 PRINTCHR$(1581)""BARCOS""BARCOS""
200 PRINT""AMIGOS""""ENEMIGOS""
210 FC=7991:FORR=1705:READR(R),SR(R)
220 PRINTCHR$(1W1:FORI=1705:PRINTTAB(1SPC1COORDENADAS"
230 PRINT"08L1SPC15W1":INPUTC
240 PRINT"1SPC131SPC1PARA1SPC1HORIZ1"PRINT"1SPC1V1SPC1PARA1SPC1VERTICAL":IN
PUTD
250 IFD="H"THEND=1:GOTO700
260 IFD="V"THENDGOSUB68:GOTO700
270 D=22
280 GOSUB740
290 X=VAL(RIGHT(C,111):V=RIGHT(W,1)
300 IFASC(V)=48ORASC(W)=57THEN720
310 Y=ASC(C)-65:1Y=Y+7THEN720
320 IFLEN(C)=12THEN280
330 F$=C+Y*22+X:FORSC=1705R(R)
340 IFPEEK(R$C)=8THEN720
350 POKES68W,7:POKES6,35
360 F$=F$+D:NEXTSC:NEXTR
370 PRINTCHR$(191:PRINTTAB(1SPC1ESTOY1SPC1COLOCANDO1SPC1)R(613SPC1BARCOS"
380 POKEV,151E=7912:NE=6:FORR=1705
390 G$=INT(RND(11)*28)+235:POKE1,C$
400 X=INT(RND(11)*81):Y=INT(RND(11)*81):D=INT(RND(11)*2)+1
410 IFD=2THEND=22
420 E$=C+Y*22+X:FORSC=1705R(R)
430 IFPEEK(R$C)=8THEN800
440 POKES6,MS=E+D:NEXTSC:MS=MS+1:NEXTR:POKE1,0
450 GOSUB760:PRINTCHR$(191:PRINTTAB(1SPC1COORDENADAS1SPC1Y2SPC1)RETURN(1SP
1PARA1SPC1D1SPARA
460 FF=7912:INPUTC=X:VAL(RIGHT(W,111):V=RIGHT(F$,1)
470 IFASC(V)=48ORASC(W)=57THEN450
480 Y=ASC(C)-65:1Y=Y+7THEN450
490 IFLEN(F$)=12THEN450
500 GOSUB760:GOSUB790:EF=FF+Y*22+X:P=PEEK(EF):B=B
510 IFP=80P=42THENDGOSUB83:POKES6,42:GOTO500
520 IFP=58THENCAC+A+1:B=1:H1=5:H5=C
530 IFP=59THENBA=B+A+1:Q=2:H1=4:H5=B
540 IFP=60THENCR=C+1:B=3:H1=3:H5=C
550 IFP=61THENSUMU=11:H1=4:H1=3:H5=W
560 IFP=62THEND=DE+1:Q=5:H1=2:H5=DE
570 DOSUB880:IFP=17THEN730
580 FORZ=1707:FORI=170500:NEXT
590 FF=7911:Y=INT(RND(11)*81):X=INT(RND(11)*81)
600 FF=FF+622*X:IFPEEK(R$X)=42THEN570
610 DOSUB790:IFPEEK(R$X)=7THENFF=X:GOSUB83:GOTO670
620 IFPEEK(R$X)=7THENPOKEFF,X,2:FF=FF+R(13):POKES63:POKEFX,42:FF=FF+1
630 IFVU=17THEN800
640 PRINT"0ANO1SPC1Y01SPC1":
645 FORI=7912705000:IFPEEK1(13)+57ANDPEEK1(13)+63THENPOKE1+W,3:POKE1,35
650 NEXT:FORI=1705000:NEXT:GOTO720
660 NEXT2:GOTO450
670 POKEFK,42:NEXT2:GOTO450
680 POKEFK,2:GOSUB830:POKEF,42
690 PRINTR(61"1SPC1TOCA80:5HHEH1:6S=Z:GOSUB819:GOSUB780
700 F$1=INT(PEEK(R$1):B1=1707THENPOKEF$1,55+4:GOSUB819:GOSUB780
710 RETURN
710 PRINT"1UISPC1GANAS1SPC1":S$=R1:GOSUB819
720 FORT=1702000:NEXT:PRINTCHR$(1471:PRINT"OTRA1SPC1VEZ1SPC18W1"
730 GETPA$:"FPA$=""THEN730
740 IFPA$="S"THENDCLR:GOTO700
750
760 FORI=799070781:POKE1,32:NEXT:RETURN
770 PRINT"AL1SPC1COORDENADA1SPC1VUELVE1SPC1A1SPC1JEPPEZAR,1:FORI=1705000:NEXT
CLR:GOTO700
780 FORI=1705000:NEXT:GOSUB760:RETURN
790 POKES2,200:FORI=157067E7,1:POKEV,1:NEXT:POKES2,0:RETURN
800 FORD=17019:FORH=1701:POKEC=H,0:B=H+1:NEXT:EC=22:ER=0:NEXT:GOTO530
810 POKEV,15:FORI=17055:FORJ=17018
815 POKES1,230:FORI=1701:NEXT:POKE1,POKES1,235:FORI=1701:NEXT
820 NEXT:FORI=1701:NEXT:POKE1,1705000:NEXT:RETURN
830 FORJ=17018:FORI=1705000:NEXT:POKE1,1705000:NEXT:RETURN
840 FORI=1705000:FORH=2760:CH=RO3,3.SUMABIN0,3.DESTRUCTOR,2

```

[illegible]

hunde uno de los barcos del ordenador, éste le indica cuál es.

Una característica interesante de este juego es la forma en que el ordenador sigue la pista de la colocación de los barcos. Se utiliza la memoria de pantalla para almacenar la información. El programa le permite al ordenador hacer un "PEEK" solamente en las posiciones que han sido alcanzadas por un disparo. Por lo tanto, el ordenador no puede hacer trampas mirando la pantalla.

Dado que los barcos tienen diferentes códigos de pantalla, el ordenador sabe cuándo un disparo ha dado en el

blanco y qué barco ha sido alcanzado. Esto demuestra que la memoria de pantalla realmente se localiza en la RAM y que puede ser utilizada para el almacenamiento de datos además de la representación de caracteres.

5K es Suficiente

Yo escribí este programa a raíz de que un amigo me dijo que no se podía hacer. El se había comprado un ordenador que no era de Commodore y para el cual estaba escribiendo una versión de este juego. Se tropezó con un pequeño problema de programa-

ción, y yo le comenté que no hubiera tenido dicho problema si se hubiera comprado un VIC-20.

El opinaba que el 5K de memoria RAM del VIC no sería suficiente para este juego. A mi se me presentó una buena oportunidad para saber exactamente la capacidad de los 3583 bytes disponibles en el VIC. Yo creo que este juego demuestra que se puede hacer mucho dentro de las limitaciones del VIC. Aunque ahora dispongo de una ampliación de memoria de 16K, sigo realizando el 90 por ciento de mi trabajo en el 5K de memoria original.

EN TARRAGONA ...

... Bits ES...

INFORMATICA PERSONAL



 **commodore
COMPUTER**

**CBM 64
VIC-20**

- * CURSOS DE PROGRAMACION
- * PERIFERICOS VIC/CBM-64

*** TODA CLASE DE ACCESORIOS, CONECTORES, ETC.
TECNHEL S.A. de Ingeniería NIF a43036094**

Compartiendo Experiencias entre amigos

Esta sección está dedicada a la colaboración de todos nuestros lectores y está dividida en dos partes:

1) Programación:

Programas y similares

2) Magia:

Trucos, sugerencias, etc.

Habrà premios y alicientes "para todos los participantes" (ver editorial página 3).

Incluimos hoy dos colaboraciones de dos chavales a ver si a los demás se les quita el miedo y nos envían cosas. Topo Loco ya apareció anteriormente en Club Commodore.

Así que: ¡animaros, chicos (...y chicas)!

Todas las colaboraciones deben venir escritas a máquina a doble espacio y los programas grabados en cinta (si es posible), o con el listado completo en impresora. Nuestros lectores más jóvenes pueden escribir a mano pero con letra muy clara.

Enviarnos vuestra dirección para que podáis poneros en contacto unos con otros.





Línea 11 y 12:

Colocan las lechugas y los bloques en su lugar.

Línea 20:

Define velocidad de juego, posición XY y la variable de color.

Línea 25 a 110:

Movimiento del topo.

Línea 120 y 121:

Comprueban qué objeto tiene delante el topo. Si es un bloque, pasa a la línea 150.

Línea 122:

El topo ha encontrado una lechuga.

Línea 123:

Caso de 0 lechugas, pasa a la línea 9000.

Línea 130:

Pone al topo en su posición.

Línea 135:

Retardo (velocidad).

Línea 140:

Vuelve a la línea 25.

Línea 150 a 168:

Explosión y un topo menos. Caso de 0 topos, pasan a la línea 9019.

Línea 3000 a 3090:

Instrucciones.

Línea 4000 a 4010:

Piden nivel dificultad.

Línea 9000 a 9012:

Música éxito (huerto vacío).

Línea 9019 a 9021:

Música de fin de juego.

Línea 9022:

Mensaje fin de juego.

Línea 20000 a 20030:

Datos musicales.

Principales variables: número de huertos (mn), número de topos (gus), número de lechugas (lech), velocidad (b), bloques adicionales (pl), posición del topo (X e Y)

Esperando noticias vuestras, se despide un colaborador.

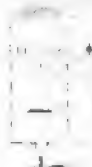
Manel

Funny Faces

Iñigo Fry
Maestro Chapi, 15
MADRID-16

14 años

Esto no os lo explico porque es tan tonto, tan tonto, tan tonto...! Que os divirtáis con los muñecos que hacen cosas.



```

10 PRINT"[CLR][90PSR0]"
20 PRINT"[10SPC][SHIFTU][40MM][SHIFTL]"
30 PRINT"[10SPC][SHIFTB][45PL][SHIFTB]"
40 PRINT"[10SPC][SHIFTB][25PC][0CSHIFTB]"
50 PRINT"[10SPC][SHIFTB][25PC][0COMM][SPC][SHIFTB]"
60 PRINT"[10SPC][SHIFTB][45PC][SHIFTB]"
70 PRINT"[10SPC][SHIFTB][SHIFTM][20MM][SHIFTN][SHIFTB]"
75 PRINT"[10SPC][SHIFTB][45PC][SHIFTB]"
80 PRINT"[10SPC][SHIFTB][SHIFTC][0MM][0COMM][SHIFTC][SHIFTB]"
90 PRINT"[12SPC][SHIFTB][SHIFTL][25PC]"
100 PRINT"[12SPC][SHIFTB][SHIFTM][25PC]"
110 FOR%0=0 TO 500
120 PRINT"[HOM][120PSR0][110PSR0]"
130 FOR%0=0 TO 200 NEXT
140 PRINT"[HOM][120PSR0][110PSR0]"
150 FOR%0=0 TO 1500 NEXT
155 GOSUB 800
160 NEXT
200 PRINT"[HOM][160PSR0][120PSR0][SHIFT][SHIFTN]"
210 FOR%0=0 TO 400 NEXT
220 PRINT"[HOM][160PSR0][120PSR0][25PC]"
230 FOR%0=0 TO 1500 NEXT
235 GOSUB 300
240 RETURN
300 PRINT"[HOM][160PSR0]"
310 PRINT"[15PC][SHIFTM][SHIFTN][25PC][SHIFTM][SPC]"
315 PRINT"[14SPC][SHIFTM][25PC][SHIFTM][SPC]"
320 PRINT"[10SPC][SHIFTU][40MM][SHIFTM][SHIFTN][25PC]"
330 PRINT"[10SPC][SHIFTB][45PC][SHIFTB][SHIFTM][25PC]"
340 PRINT"[10SPC][SHIFTB][25PC][25PC][SHIFTB][SPC][SHIFTB]"
350 PRINT"[10SPC][SHIFTB][25PC][0COMM][SPC][SHIFTB][0CSHIFTB]"
360 PRINT"[10SPC][SHIFTB][45PC][SHIFTB][25PC][SHIFTB]"
370 PRINT"[10SPC][SHIFTB][SHIFTM][20MM][SHIFTM][SHIFTB][25PC][0COMM]"
375 PRINT"[10SPC][SHIFTB][45PC][SHIFTB][SPC][SHIFTM][25PC]"
380 PRINT"[10SPC][SHIFTB][SHIFTC][0MM][0COMM][SHIFTC][SHIFTB][25PC]"
390 PRINT"[12SPC][SHIFTB][SHIFTL][0COMM][SHIFTM][25PC]"
395 PRINT"[12SPC][SHIFTB][SHIFTM][25PC]"
400 FOR%0=0 TO 700 NEXT
410 PRINT"[HOM][160PSR0]"
420 PRINT"[110PSR0]"
510 PRINT"[15PC][40MM][45PC]"
515 PRINT"[15PC][0COMM][25PC][0COMM][45PC]"
520 PRINT"[0SPC][0COMM][0COMM][SHIFTL][20MM][SHIFTM][0COMM][0COMM][SPC]"
530 PRINT"[10SPC][SHIFTB][45PC][SHIFTB][25PC]"
540 PRINT"[10SPC][SHIFTB][25PC][0SHIFTB][25PC]"
550 PRINT"[10SPC][SHIFTB][25PC][0COMM][SPC][SHIFTB][25PC]"
560 PRINT"[10SPC][SHIFTB][45PC][SHIFTB][25PC]"
570 PRINT"[10SPC][SHIFTB][SHIFTM][20MM][SHIFTM][SHIFTB][25PC]"
575 PRINT"[10SPC][SHIFTB][45PC][SHIFTB][25PC]"
580 PRINT"[10SPC][SHIFTB][SHIFTC][0COMM][SHIFTC][SHIFTB][25PC]"
590 PRINT"[12SPC][SHIFTB][SHIFTL][25PC]"
600 PRINT"[12SPC][SHIFTB][SHIFTM][25PC]"
610 RETURN

```

MS

MicroSistemas

Relación de Tiendas en Madrid donde se Vende ya MicroSistemas

COMPU-SHOP	Infanta Mercedes, 89
COMPUSTORE	Doce de Octubre, 32
COMPUSTORE	Sor Angela de la Cruz, 28
COSESA	Barquillo, 25
DIDISA	Pintor Rosales, 26
ELECTRONICA SANDOVAL	Sandoval, 3, 4, 6
MICROTEC	Duque de Sesto, 30
PEÑA, Librería	Cavanilles, 52
SINCLAIR STORE	Bravo Murillo, 2
VIDEO MUSICA	Orense, 28



Tengo un micro ordenador Commodore 64 y el Floppy 1541.

Así mismo les ruego me proporcionen información sobre los programas EASY SCRIPT y CALL RESULT, y cualquier otro programa de proceso de datos y sobre idiomas, siempre refiriéndome para diskette.

JESUS UGARTE UGARTE,
LEON Y CASTILLO, 228 ATICO X,
LAS PALMAS-4

En este número publicaremos reseña sobre EASY SCRIPT.

Ver galería de Soft (página 2), en el próximo publicaremos un amplio artículo sobre el mismo. Tomamos nota sobre CALL RESULT.

Sería interesante que la revista no se tecnicase excesivamente pues se convertiría en incomprensible para la mayoría de los usuarios de Commodore que creo somos simples aficionados y no técnicos, también sería bueno para nosotros que se tocasen algunos POKES de direcciones de memoria en cada revista para que se fuese desvelando este verdadero misterio para los aficionados, lo que podía hacerse con pequeños programas que nos mostrasen su aplicación prácticamente.

Sería conveniente también que se explicasen paulatinamente los conceptos de buffer, puntero, vector, memoria intermedia de teclado..., etc.

Nada más, reciban un cordial saludo de su nuevo suscriptor.

Según podrás apreciar, intentamos dar un poco para todos, tanto para los novicios como para los muy veteranos e incluso para los niños (a partir del próximo nº 3).

Tomamos nota para comenzar en nuestro próximo con un glosario de términos y con "el misterio de los Pokes". De hecho nos gusta el término como título. Te lo robamos.

Con esto contestamos también a JOAQUIN LOZANO DE VALENCIA que nos pide un estudio de la diferencia del POKE entre el VIC y el 64. Así como a varios lectores que nos han llamado con la misma petición.

CONTRABANDO

Estimados amigos:

Me dirijo a vosotros porque en fecha no muy lejana voy a comprarme un cassette Commodore para poder grabar mis propios programas, he preguntado el precio pero nadie parece saberlo, por eso os escribo esta carta.

Ya que vosotros o ustedes lo fabricáis, deberíais saber el precio exacto en el mercado.

También desearía saber, si es posible el precio en Andorra. Muchas gracias.

Nota: Me ha gustado mucho la editorial del nº 1 de la revista Commodore World.

JULEN ARRUNTIA
C/ PIO XII, 13, Esc. B 2º C
PAMPLONA

El VIC vale 34.900, el C-64 79.900 y el Cassette 12.000. Hay una casa de Zaragoza B.M. que se anuncia aquí página 21. Llámale.

Oye majo..., eso de Andorra..., ¡no pretenderás que te ayude al contrabando!

RESPECTO A LA PROPIEDAD, ¿CUANDO?

Queridos amigos:

Ante todo, felicitaros por esta nueva revista que viene a llenar un hueco muy importante para los usuarios de Commodore, ya que, a pesar de que la revista del Club era muy interesante, no cubría todo lo que ésta desde su primer número promete; tengo un 64 desde hace pocos meses y toda la información me parece poca, así que mi felicitación de nuevo y mis mejores deseos de éxito que no dudo tendréis.

Con mucho gusto me suscribiría pero el buzón de casa es lo bastante pequeño para hacerme desistir pues desaparecen o los estropean intentando meterlas, de todas formas os prometo ser un lector asiduo.

Os acompaño unas pequeñas experiencias mías que pueden encajar en la sección de MAGIA y en caso de que mereciera la pena publicarlas, os seguiría mandando.

DOMINGO MARQUEZ GALLEGO
PARQUE VOSA, 3, 4º C
MOSTOLES (MADRID)
TELEF. 614 21 48

Gracias por tus palabras. Pasamos tu receta a MAGIA. ¡Ah! y síguenos mandando cosas.

¿CLUB DE INFORMÁTICA?

Somos suscriptores suyos, y a la vez hemos organizado un Club de Informática, donde empleamos ordenadores Commodore 64 y Vic-20.

Les enviamos folleto del primer curso que hemos organizado y rogamos nos tengan al corriente de cuanta información consideren que nos sea de utilidad.

JAVIER GARCIA DIAZ

BLANCA

A través del folleto no estamos muy seguros si se trata de un Club o de una academia. Tampoco se dice cuál es la diferencia entre socios y no socios ni qué se necesita para ser socios. Ampliad esta información, por favor. Si sois Club os incluiremos en el próximo número en MARKET CLUB.

TODO DE TODO

Hola, amigos:

Os escribo atendiendo a la petición de sugerencias y opiniones que hacéis en el número 1 de Commodore World, y la impresión que he sacado después de leerlo supongo que es la misma de todos los demás suscriptores:

¡MAGNIFICA!

Me parece fenomenal, por ejemplo, que incluyáis cada mes un tema, como el de generación de caracteres, en el número 1, acompañado de un programa y explicado exhaustivamente, con ejemplos, ilustraciones, etc...

A este respecto, hay algunos temas que me gustaría ver desarrollados en próximos números, como puede ser una explicación detallada de las rutinas de KERNAL, su funcionamiento, utilidad, modo de introducirlos en un programa, etc., todo ello acompañado de un programa explicado línea a línea.

Otro tema que me gustaría dominar también es el de la composición de música con el conjunto de comandos del SUPEREXPANDER; podéis explicar, por ejemplo, cómo pasar de la música escrita en pentagrama con corcheas y demás símbolos musicales a la notación del cartucho. También sería interesante que de vez en cuando se incluyera en la revista una canción popular, un efecto de sonido, etc..., para usar con el cartucho, todo ello debidamente explicado.

Habréis observado que hago mucho hincapié en eso de explicar lo que decís en la revista, y es que mi experiencia en la etapa anterior de la revista era a veces de quedar defraudado cuando se tocaba un tema y tenía que usar esa información sin comprenderla (p. ej. cuando salió el programa en que aparecía BUG en TV, no entendía por qué tal variable tenía que tomar tal valor, etc...).

También me interesa mucho el problema de cómo borrar parte de un programa que sólo se ha de utilizar una vez en él, y así recuperar esta memoria para variables... Esto ya se tocó anteriormente, pero intenté aplicarlo a un programa mío, sin entender



cómo lo hacía, y no dio resultado, por eso os pido que no dejéis nada sin comentar, por sencillo que parezca, en la sucesiva.

Pasando a otro orden de cosas, me gustaría mucho que se publicara un curso de programación en lenguaje máquina, partiendo de nivel cero, y que incluyera programas explicados, descripción de comandos, etc...

Os agradecería que informara en la revista, o a mí personalmente, sobre la existencia o no de algún curso en lenguaje máquina editado en español, pues, que yo sepa, no existe ninguno, a pesar de la gran demanda que debe de haber.

Como ya he visto que en el número 2 vais a publicar un artículo sobre la creación de nuevos mandatos al VIC, no os pido nada por ahora en este sentido, pero sí me gustaría saber cómo se hace para cambiar el comando LIST por LISTA, o READ por LEE, etc..., que tengo entendido que se puede hacer, y espero que lo publiquéis próximamente.

Siguiendo con la opinión que pedis en la editorial del número 1, os diré que me parece fenomenal que se incluyan varios juegos en cada revista, ya que un mayor nivel de profesionalización de la revista no debe estar reñido con la dedicación de parte de ella al esparcimiento y la distracción.

Una buena idea por vuestra parte ha sido la de hacer esos sorteos de software periódicamente, creo que esto potenciará más la participación de los usuarios.

Bueno, por ahora no os voy a dar más el tostón, espero poder mandar alguna colaboración un día de estos. ¡Hasta pronto!

FRANCISCO SAEZ SOTO
C/ VIRGEN DEL ROSARIO, 28
ALCANTARILLA
MURCIA

¡Tú te quedas solo pidiendo!

Muchas cartas así y ya tenemos ideas editoriales para todo el año.

Me temo que nosotros tampoco conocemos nada editado aquí sobre lenguaje máquina.

Los otros puntos que mencionas se irán tocando a lo largo del año.

Mientras tanto, incluyo tu dirección por si algún lector quiere cartearse contigo sobre estos temas.

Paso tus recetas a MAGIA.

¡Ah! y gracias por los piropos.

En los números 11, 12 y 13 de Club Commodore salieron unos artículos sobre este tema. Pretendemos comenzar un curso paso a paso en el próximo número. Asimismo apareció un mapa de memoria para el Vic 20 en el nº 1 en el que faltaban las siguientes posiciones:

5030C 780 almacenamiento temporal de A durante SYS.

5030D 781 almacenamiento temporal de X durante SYS.

5030E 782 almacenamiento temporal de Y durante SYS.

50304 783 almacenamiento temporal de F durante SYS.

En el nº 7 apareció el mapa de memoria para el C-64. En el nº 15 referente a las tablas de KERNAL

apareció un artículo con unas tablas. En el futuro (no prometo en qué número) trataremos este tema más a fondo con ejemplos prácticos.

NUEVO CLUB

Hace poco tiempo quedó establecido y reglamentado el Club de Programación Alaiz, en Pamplona. Está dirigido a estudiantes de B.U.P. y C.O.U. de esta ciudad, y el fin que tiene es la introducción en el mundo de la informática y programación BASIC. No tiene, por lo demás, ningún fin lucrativo. El club propiamente posee equipos COMMODORE 64, VIC-20 y diversos periféricos (impresora, unidad de disco y cassette).

Así mismo, dado que en Pamplona los distribuidores no tienen una gran información de bibliografía Commodore, periféricos y Software, y dado que formamos un club de programación me gustaría nos enviase amplia información en estos campos.

Las actividades que realizamos requieren esta información ya que dentro de los chicos que participan los hay ya adelantados en programación.

JESUS RAFAEL MARTINEZ

ESTAREMOS EN INFORMAT

Hola amigos de COMMODORE WORLD: Me llamo Ramón y esta Navidad me regalaron un COMMODORE VIC-20, incluido el "DATASSETTE". Yo tengo una gran afición a los "Videojuegos" y cuando mi hermano me compró el "Micro-Sistemas" Nº 7, con el cual regalaban vuestro primer número, me paré al leer la editorial ya que en el apartado de "Futuro" había una cosa que me interesaba, las consultas, como decía me paré para escribir esta carta, puesto que me gustaría que me enviarais o publicarais en vuestra/nuestra revista información sobre todos los "Video-juegos" disponibles para el VIC-20, sean en cartucho o cassette.

Supongo que tendréis un "Stand" en el "INFORMAT-84", me pregunto si venderéis "Video-juegos", "Joysticks", "Paddles", cartuchos diversos, etc. Por favor contestadme lo antes posible, sea en la revista o por carta, adiós amigos.

Gracias por haber leído esta carta y hasta pronto.

RAMON PLANS ESPERABE
C/ CORTS CATALANES, Nº 6, 3º-4.º
CALDES DE MONTBUI
(BARNA)

Bienvenido a la familia Commodore, Ramón. Estaremos en Informat-84 por supuesto, pero compartiendo el "stand" bien con la editorial de MicroSistemas o con Microelectrónica y Control, todavía no está decidido. Nosotros personalmente no venderemos Soft excepto las cintas de programas de la revista, pero seguro que habrá bastantes distribuidores de productos Commodore que muy probablemente lo anunciarán en nuestras páginas de abril. Repito tu petición sobre soft en la columna correspondiente.

ESTIMADOS AMIGOS:

Por la presente os remito el "BOLETIN DE SUSCRIPCION" para la "nueva" Revista y unas dudas por mi parte. En el mes de julio pasado comencé mi suscripción a la Revista "CLUB COMMODORE" y por lo tanto he recibido 6 números de dicha revista y el primer número de esta nueva, por lo que aún me quedan 4 números para acabar con la antigua suscripción, pero no he visto que hagáis ninguna referencia a la situación en la que quedamos los "antiguos" suscriptores, es decir, ¿recibiremos la nueva revista igualmente sin mandar la nueva suscripción hasta que la anterior finalice? En este caso el nuevo Boletín que os adjunto podéis guardarlo hasta que acabe mi suscripción o remitírmelo de nuevo.

Me gustaría me permitierais que haga algún comentario respecto a los artículos y notas a los programas que han aparecido en "CLUB COMMODORE" y en "COMMODORE WORLD".

Creo que en algunos de éstos se dan largas explicaciones, algo muy interesante, pero esto ocurre casi siempre con las líneas de programa en "Basic" o programas escritos en este lenguaje. En cambio cuando se refiere a algún programa en código máquina las referencias son mínimas y las explicaciones son del todo un jeroglífico para mí, puesto que aún no he encontrado ningún manual que explique con detalle, como podría serlo para el "Basic", cómo utilizar el código máquina, por qué y cuándo. Sería de desear por mi parte hubiera una sección en la nueva revista tal como "CLAVE" en la que se enseñe o se explique de alguna manera cómo programar en código máquina.

Y para finalizar esta, demasiado larga, carta, felicitéis de mi parte a Don Alfonso Izquierdo Font por su excelente y sencillo programa "DIBUVIC" que, aunque un poco corto de explicaciones para poder sacarle el mayor provecho, para los que no entendemos de Geometría, resulta muy interesante.

FELICIDADES Y MUCHA SUERTE A TODO EL "EQUIPO" de COMMODORE WORLD.

RAMON RONCERO SANCHON
C/ JACINTO VERDAGUER, 36
MADRID-19

¡Gracias!

Aprovechamos tu carta para avisar a todos los "viejos" suscriptores que andan un poco despistados como tú. Todo los antiguos suscriptores pasan automáticamente a ser suscriptores de Commodore World hasta que finalice su tiempo de suscripción. Si alguno nos ha enviado una suscripción duplicada que nos lo comuniqué, por favor para devolverla o dejarla en depósito para renovación, a elegir.

Sobre lenguaje máquina te digo que lo mismo que a Francisco Sáez. Nos meteremos con ello lo más pronto posible.

Alfonso Izquierdo, un suscriptor, recoge tu felicitación desde estas páginas.

Pasamos tu pregunta a SEAMOS PREGUNTONES.



PREGUNTONES

Me gustaría me contestaseis al siguiente problema: Tengo un VIC-20 al que amplío la memoria con un cartucho de 8K RAM y un SUPER-EXPANDER colocados en el "VIC-20 Expander" de INDESCOMP (de hasta cuatro cartuchos), pero ocurre lo siguiente: 1.—En el caso de conectar los dos cartuchos la memoria RAM libre final que aparece en pantalla no es el resultado de la suma de los 8K RAM el SUPER EXPANDER y el propio equipo base. 2.—Si conectados los dos cartuchos y cualquiera de ellos se desconecta mediante el interruptor del "Expander" o lo quito de este (siempre con la alimentación desconectada) el resultado una vez vuelta a conectar la alimentación, es que una vez aparecido en la pantalla la "presentación", automáticamente desaparece como si se hubiese teclado RUN/STOP y RESTORE y cada vez que se escribe algo, ocurre lo mismo. 3.—Sucede que en programas largos en los que se utiliza el modo GRAPHIC 2 aparece OUT of MEMORY pero consultando FRE(x) todavía queda memoria y si se elimina la instrucción GRAPHIC 2, el programa se ejecuta hasta el final, pero claro, sin alta resolución.

¿QUE HACER?

RAMON RONCEROS SANCHON
C/ JACINTO VERDAGUER, 36
MADRID-19

1º Cuando se utilizan SKRAM (o más) con el Super-Expander, los 3K RAM de este último están disponibles pero no para BASIC, sólo para lenguaje máquina.

2º Comprueba que estén bien introducidos los cartuchos en sus conectores.

El fallo que dices puede ocurrir si el expansor no dispone de resistencia de "Pull-up" asociadas a los interruptores (¿Fallo de diseño...?), pero no puedo asegurarlo ya que Indescomp no nos ha enviado ninguna muestra de sus productos.

3º El error "OUT of Memory" puede ser debido a la existencia de algún "GOSUB" sin "RETURN". Aunque disponga de memoria suficiente el stack puede llenarse dando lugar a este error.

Prueba este programa:
10 GOSUB 10
y consulta: FRE(X)

ATENCION DISTRIBUIDORES

Los siguientes lectores nos piden información sobre soft y productos Commodore.

Software - Educación y Negocios,
José Antonio Herrera Arranz, Bristol Mayer, Isla de Java, 1, Teléfonos: 729 48 88 (Despacho) - 705 52 49

Me gustaría que me enviarais o publicarais en vuestra/nuestra revista información sobre todos los "Videojuegos" disponibles para el VIC-20, sean en cartucho o cassette.

Ramón Plans Esperabe. C/ Corts Catalanes, nº 6, 3º-4º. Caldes de Montbui (Barna).

Así mismo, dado que en Pamplona los distribuidors no tienen una gran información de bibliografía Commodore, periféricos y Software, y dado que formamos un club de programación me gustaría nos enviasen amplia información en estos campos.

Las actividades que realizamos requieren esta información ya que dentro de los chicos que participan los hay ya adelantados en programación.

Club Alaiz. Plaza Monasterio Santa Gemma, s/n. Teléfonos: 25 44 80 - 25 77 04. Pamplona.

Más información de otras publicaciones y programas establecidos.

Tengo un C-64

Juan López Ferrer, Plaza de Careaga, 9, 3º C. Teléfono: (951) 23 26 09. Almería.

Agradecería información sobre la existencia de cartuchos en el mercado para el C-64.

Jesús J. Inserte Peralta. C/ Columbario, 4. Valencia-20.

Me interesaría si ello fuera posible el que me remitierais un catálogo/lista de precios de los posibles accesorios o complementos que se pueden disponer para un VIC-20. También bibliografía en España traducida.

José Luis Lozano. Rambla Justo Oliveras, 81, 4º, 1º. Hospitalet de Llobregat. Barcelona.

Les rogamos que si tienen algún libro sobre programas, nos lo comuniquen, tipo de programas, precio y demás.

Ismael Iglesias Filgueira. Dirección de envío: Calle Arco, 9 y 11 - 2º E. La Coruña-3.

Os rogaría me enviarais una lista de publicaciones (libros, revistas, etc...) en castellano, sobre el COMMODORE 64.

Román Elizalde. Eichstrabe, 46. 3000 Hannover-1


Nos da casi vergüenza tener que admitir que estamos casi en las mismas condiciones que vosotros. Desde aquí hacemos una llamada a todos los distribuidores para que nos envíen listas de soft y productos disponibles con precios.

Desde ya, podéis poneros en contacto con las firmas que se anuncian en la revista. Ellas os ayudarán sin reservas.

Al publicar vuestra dirección los distribuidores pueden enviarnos también información directamente y, por favor remitidnosla en bien de todos.

DISTRIBUIDORES

Si queréis vender vuestros productos no hagáis un misterio de ellos. "El buen paño en arca se vende", pertenece a la prehistoria.



*La MAGIA son trucos, la MAGIA
es divertida.*

*La MAGIA es hacer lo que nadie
se ha atrevido.*

Magia

*La MAGIA es una columna mensual llena de
consejos, trucos, de esto y aquello del mundo del
software, hardware y aplicaciones.*

*Cada mes, MAGIA les trae trucos breves y
útiles de informática procedentes de todo el
mundo - trucos descubiertos por los demás que
hacen que la informática sea más fácil, más
divertida o más animada.*

*MAGIA habla de ideas sencillas, programas
de una sola línea, subrutinas útiles, hechos de
informática poco conocidos y otras cosas de
interés. Buscamos material nuevo o renovado
que resulta ser de valor actual para usuarios de
equipos Commodore y que puede utilizarse con
un mínimo de tiempo, esfuerzo o conocimientos
teóricos.*

*MAGIA resulta ser la fuente más completa de
información para la informática práctica,
además de ser un foro internacional para
compartir trucos con otros aficionados. Envía
tus trucos a:*

COMMODORE WORLD

Pedro Muguruza, 4

Madrid-16

*La revista "Commodore World" sorteará seis
paquetes de software
en julio y diciembre entre todas las
contribuciones publicadas.*

DETECTAR PULSACIONES

Cuando se utiliza una sentencia GET para detectar una pulsación de tecla, el hecho de que se hayan salvado las pulsaciones anteriores en el "keyboard buffer" (memoria tampón del teclado) podría constituir un problema. A menudo, cuando se termina el juego, el jugador tiene que volver a iniciar el juego mediante la pulsación de una tecla. A continuación, presentamos la forma habitual de hacerlo:



```
510 PRINT "PULSA TECLA PARA  
CORRER"  
520 GETA$:IF A$="" THEN 520  
530 RUN
```

Estas líneas volverán a ejecutar el programa incluso si una tecla fue pulsada *antes de que se ejecutara la línea 510*. El problema se solucionará si se añade la línea 500 *FOR I=*

=1 TO 10: GETA\$: NEXT.

Se puede hacer lo mismo con **una sola** línea si las líneas 500-530 se sustituyen por la siguiente:

```
500 PRINT "PULSA CUALQUIER TECLA PARA  
CORRER": POKE 198, 0: WAIT 198, 1: RUN.
```

El comando "POKE 198, 0" borra el "keyboard buffer". "WAIT 198, 1" le indica al ordenador que espere hasta que se pulse una tecla.

Westmoreland Commodore Newsletter

COMILLAS

Cuando se utiliza la sentencia Print con la información entre comillas, muchas veces es aceptable omitir el segundo par de comillas. Por ejemplo, el ordenador tratará las siguientes dos sentencias de la misma forma:

```
100 PRINT "MAGIC IS FUND"  
110 PRINT "MAGIC IS FUND"
```

Si se elimina el segundo par de comillas, se ahorra un byte de memoria, una pulsación y un espacio en la línea de pantalla, lo cual puede ser importante a veces. Pero hay que tener cuidado la última letra de la información contenida entre comillas **debe** ser lo último de la línea del programa. Dada la presencia del comando GOTO en esta sentencia, el segundo comando, 120 PRINT "ABRACADABRA": GOTO 120, no puede ser eliminado.

Dada la presencia del punto y coma después de esta sentencia Print, hace falta el comando de cierre, 130 PRINT "LEGERDEMAIN".

L.F.S.

PUNTO Y COMA

Muchas veces se pueden eliminar los puntos y comas entre varias unidades de información que van a ser impresas en la misma línea. Si no existe ninguna duda sobre la posición final de una unidad de información y la posición inicial de la siguiente, los puntos y comas son innecesarios. En este ejemplo:

```
140 PRINT A$;B$;C$;D;"E"
```

una unidad de información se distingue perfectamente de otra mediante el uso de los signos de dólar y las comillas.

Esta línea puede ser abreviada de la siguiente forma:

```
140 PRINT A$B$C$D"E"
```

El punto y coma **debe** ser incluido en esta línea:

```
150 PRINT F;G
```

Si se suelta, el ordenador imprimiría innecesariamente el valor de la variable FG.

Jacinto Arroyo (Badajoz)

COMAS

Si se coloca una coma entre dos unidades de información en una sentencia Print, la segunda unidad se imprime en la siguiente posición de TAB previamente establecida en la



pantalla. En el Commodore-64, existen cuatro posiciones de TAB por cada línea de pantalla, mientras que en el VIC-20, sólo hay dos. Las comas adicionales colocadas entre las unidades de información hacen que las posiciones adicionales de TAB se salten. La A y la B se imprimen en las posiciones TAB 1 y 4 de la siguiente forma:

```
100 PRINT A,,B
```

Pedro D'Asilva (Lisboa)

FORMACION DEL MARCO EN PANTALLA

Esta rutina imprime un marco alrededor de la pantalla del Commodore PET, pero también funciona para el C-64. Si se cambian unos cuantos números, también funcionará para el VIC-20.

```
3000 REM **RUTINA DEL MARCO**  
3010 PRINT "(clear)"; FOR I=1 TO 39:  
PRINT F$; NEXT: PRINT "(cur-  
sor up)"  
3020 FOR I=1 TO 23: PRINT F$TAB  
(38)F$; NEXT I  
3030 FOR I=1 TO 39: PRINT F$; NEX-  
T I: PRINT "home"
```

En esta rutina, F\$ puede representar cualquier carácter. El usuario puede sustituir F\$ por su carácter preferido (entre comillas, por supuesto) o cambiar F\$ cada vez que se dibuja el marco. Por ejemplo:

```
40 F$="X": GOSUB 3000.
```

Pero es importante recordar que si esta rutina se utiliza como subrutina, hay que añadir una línea 3040 Return, y hace falta una sentencia End en algún sitio antes de la rutina para que el programa no entre en la rutina mientras éste se está ejecutando.

Mercedes Antón (Madrid)

CONTROL PROGRAMABLE DEL CURSOR

El cursor puede ser colocado en cualquier posición de pantalla mediante el uso de una rutina como la que se presenta a continuación:

```
10 X$="(39 crsr rights)": Y$="(24 crsr downs)"  
100 X=20: Y=10: GOSUB 3000  
110 PRINT "SORCERY"  
2999 END  
3000 REM **POSICIONAR CURSOR**  
3010 PRINT "(home)" LEFT$(X$, X) LEFT$(Y$, Y):  
RETURN
```

Las variables X\$ e Y\$ deben de inicializarse al principio del programa y no volver a cambiarse. El código en la Línea 100 establece la posición deseada del cursor, y a continuación coloca el cursor en la columna X y la línea Y. (X=0 para la columna de la extrema izquierda, Y=0 para la línea superior). Al volver de la subrutina, la línea 110 impide la información deseada en dicha posición. La línea 2999 prohíbe la ejecución no deseada de la subrutina.

Nuestro experto añade: Otro modo de posicionar el cursor es, POKE 781, línea: POKE 782, columna: POKE 783, 0: SYS 65520.

Begoña Garrido (Bilbao)

ADICIONAL CONTROL PROGRAMABLE DEL CURSOR

Una vez asimilada la técnica del truco anterior, se puede abreviar mucho mediante la eliminación de la línea 10 y la modificación de la línea 3010 para que quede de la siguiente forma:

```
3010 PRINT "(home)" LEFT$ ("(24 crsr downs)", Y) TAB  
(X); RETURN.
```

Ignacio Casanova (Tarragona)



CONTAR EN HEXADECIMAL

Si se cuenta en hexadecimal resulta más fácil acostumbrarse al sistema de numeración hexadecimal. El "signo de libra" o el "signo de números" indica que lo que sigue es un número, mientras que el signo de dólar constituye una costumbre ampliamente usada que indica la notación hexadecimal. También existe una costumbre para indicar la notación binaria, es decir el signo de porcentajes, pero no se utiliza con mucha frecuencia.

Javier Roman (Zaragoza)



TRUCOS DE UNA SOLA LINEA

Este truco de una sola línea convierte hexadecimal en decimal.

Convierte un número hexadecimal de cuatro dígitos, expresado como una variable de serie H\$, en su equivalente decimal, expresado como la variable numérica D. Se presenta a continuación:

```
100 D=0:FOR I=1 TO 4:D%=ASC(H$):D%=48+(D% 64)*7:
H$=MID$(H$,2):D=16*D+D%:NEXT
```

Para ver cómo funciona la rutina, se añaden las dos siguientes líneas, y se ejecuta el programa:

```
50 INPUT "HEX":H$
150 PRINT D
```

Margaret Ittel (Washington)

OTRO TRUCO DE UNA SOLA LINEA

Este truco convierte el número decimal D en su equivalente hexadecimal H\$ de cuatro dígitos:

```
200 H$="":D=D/4096:FOR I=1 TO 4:D%=D:H$=H$+CHR$(
48+D%-(D%>9)*7):D=16*(D-D%):NEXT
```

Este programa de conversión se puede comprobar si se añade, además de 250 PRINT H\$, al programa en el truco anterior.

A.W. Grym (New York)

"POKES" DE COLORES

Es fácil recordar los valores de los "Poke" para los primeros ocho colores del VIC 64 - el valor del "Poke" es uno menos que el número que aparece en la tecla de color. BLK se encuentra en la tecla 1, así que el valor "Poke" es el 0; WHT se encuentra en la tecla 2, y el valor del "Poke" es el 1 etcétera.

L.F.S.

CARACTERES PROGRAMADOS

En el artículo sobre la creación de caracteres que aparece en el nº 1 de "Commodore World", se dice en la primera columna de la página 10 que se pierde la capacidad de utilizar el juego de caracteres pre-programado del VIC al crear caracteres nuevos en un programa y modificar los punteros para que apunten a la RAM (o así me lo parece, al menos), pues bien, la experiencia que yo tengo al respecto es la de estar un día probando un programa de juegos y empezar a hacer POKES en la posición 36869 desde 255, que es el nº necesario para que el VIC leyera los caracteres programados del juego, hacia abajo; al llegar a 240, vi que los caracteres que tenía en pantalla pasaban de los programados a los normales del VIC, lo que comprobé después incluyendo este comando POKE 36869,240 en el programa y haciendo que un PRINT pusiera en pantalla letras cuyo código había modificado con anterioridad, lo que supongo que significará que de nuevo estaba leyendo en ROM.

Francisco Sáez Soto

REPETICION DE TECLA

"En el COMMODORE 64, sólo las teclas de movimiento de cursor, la de INST/DEL y la barra espaciadora presentan autorrepeticion cuando se las mantiene pulsadas. Si se activa el bit 7 del registro 650, todo el teclado presentará esta característica. Para ello, basta teclear: POKE 650,128".

Rafael García Segura (Málaga)

HELICOPTERO

La que doy a continuación es una subrutina que tiene por misión simular el sonido de un helicóptero. Utiliza el comando SOUND, es decir, que sólo se puede ejecutar en presencia del SUPEREXPANDER. La variable JJ no tiene ningún efecto sobre el sonido, sólo determina el tiempo que está ejecutándose la subrutina.

```
10 FOR JJ=1 TO 20
20 FOR J = 60 TO 65
30 SOUND 128+J,0,0,15-INT (J/5)
40 SOUND 0,255-J,0,0,1+INT (J/5)
50 SOUND 0,0,128+J,0,15-INT(J/5)
60 SOUND 255-J,0,0,0,1+INT (J/5)
70 NEXT J
80 NEXT JJ
90 SOUND 0,0,0,0,0
```

Francisco Sáez Soto

FUNCIONES ESPECIALES DEL TECLADO

Introduciendo directamente o en programa —POKE 650,255— se consigue que todo el teclado se vuelva repetitivo. —POKE 650,0— lo retorna a la normalidad.

Pulsando la tecla Control y al mismo tiempo S, pone el cursor al principio de la pantalla. Pulsando Control T: Delete. Control Q: Cursor abajo. Control % : Cursor derecha. Control =: Pone el cursor del 1. fondo. Control E: Lo retorna a su color. Control N: Activa minúsculas. Control H: Fija Mayúsculas o minúsculas. Control I: Suelta mayúsculas o minúsculas. Control M: Return.

Esto se puede utilizar directamente o en programa. Aunque son repetición de las funciones que conocemos, algunas veces pueden ser útiles. Pulsando Print "N control N sale N en inverso. Al ejecutar "Run" tendremos las minúsculas, es más cómodo que POKE o que CHR\$ y más rápido.

Domingo Márquez

COMO SIMULAR LA INSTRUCCION "PRINT AT"

Muchos usuarios del VIC-20 y del COMMODORE 64 echan de menos la instrucción

PRINT AT (coordenada vertical), (coordenada horizontal) que otras versiones de BASIC sí poseen. Esta instrucción permite posicionar el cursor en cualquier punto de la pantalla, tras la mera indicación de la fila y la columna en que deseamos situarlo. Una pequeña subrutina de una sola línea logra incorporar esta facilidad a cualquiera de los dos equipos arriba mencionados: basta con introducir el número de fila en la variable FILA, introducir el número de columna en la variable COL, y acudir a la subrutina que a continuación transcribo:

```
10000 POKE 781,FILA:POKE 782,COLUMNA:
POKE 783,0:SYS 6520:RETURN
```

Una vez incluida esta subrutina en nuestro programa, para colocar el cursor —por ejemplo—, en la columna 16 de la fila 9, escribiremos simplemente:

```
FILA=9: COL=16: GOSUB 10000
```

Mágico, ¿verdad?

Angel-Francisco Alegre Amor. C/ Dr. Marañón, 2, 6C.
CACERES. Tfno.: (927) 24 98 85

FEC.	ASTO	PT	CONCEPTO	CUENTA	DESCRIPCION	DEBE	HABER
0101	1	3	REPORTACIONES	570000	CAJA	100000.00	0.00
0101	1	3	REPORTACIONES	100000	CAPITAL		100000.00
0101	1	3	*IMP. A-1, P-2	100000	CAPITAL		300000.00
0101	1	2	FOTOCOPIAR	570000	CAJA	150000.00	
0101	2	2	FOTOCOPI	570000	CAJA		50000.00
0101	2	3	LETRA ACPT.	450000	EFFECT.COMER. A PAGAR		100000.00
0101	3	1	DE CASH	572000	BANCO A	500000.00	
0101	3	2	ING. BANCO A	570000	CAJA		500000.00
0101	4	1	VARIOS	570000	MATERIAL DE	300.00	
0101	4	2	MAT. OFICINA	570000	CAJA		15000.00
0101	5	1	MOBILIARIO	200000	OTRO INMOV.		
0101	5	2	TAMBI	572000	BANCO A		
0101	5	3	ACPT. LETRA	450000	EFFECT.COM		
0101	5	4	ACPT. LETRA	450000	EFFECT.COM		
0201	6	1	PRODUCTO A	600000	OTROS TO		
0201	6	2	SU FRA	400000	PROVEEDOR		
0201	7	1	PRODUCTO A	600000	COMPRA A		
0201	7	2	TRANSPORTE	450000	OTROS TO		
0201	7	3	SI FRA	400000	PROVEED		
0201	7	4	TRANSPORTE	570000	CAJA		
0201	8	1	VENTA CONTAD	570000	VENTAS		
0201	8	2	PRODUCTO A	700000	VENTAS		
0201	9	1	ORDENADOR	200000	INMOVIL		
0201	9	2	TAMBI	172000	BANCO A		
0201	9	3	ACPT. LETRA	450000	EFFECT		
0201	10	1	PAPEL LIST.	660000	MATE		
0201	10	2	PAPEL LIST.	570000	CAJA		
1001	11	1	*IMP. A-3, P-1	570000	CAJ		
1001	11	2	*IMP. A-3, P-2	100000	VEN		
2901	12	1	ENERO	640000	BA		
2901	12	2	RO. ALQUILER	572000	BA		

casa
 de
 software so

c/ aragon, 272 8º 6º
 tel 215 69 52
 barcelona - 7

EMPRESA CASA DE SOFTWARE, S.A.		DISCO	1 FECHA	16-02-84	HOJA: 1
CUENTA DESCRIPCION	DEBE	HABER			
300000 COMERCIALES	0.00	0.00			
310000 PRODUCTOS TERMINADOS	0.00	0.00			
350000 MATERIAS PRIMAS Y AUX.	0.00	0.00			
400000 COSTO A	450000.00	0.00			
601000 MATERIA A	0.00	0.00			
630000 I.T.E.	0.00	0.00			
640000 APRENDIZAJOS	20000.00	0.00			
650003 OTROS TRANS. V FLETES	10000.00	0.00			
650000 MATERIAL DE OFICINA	20000.00	0.00			
650001 COMUNICACIONES	0.00	0.00			
650002 PUBLICIDAD Y PROPAGANDA	0.00	0.00			
650005 VARIOS	0.00	0.00			
650000 VENTAS A	0.00	50000.00			
650006 COMISIONES A	0.00	0.00			
	0.00	0.00			
		141000.00			

casa
de
software sa
c/argos

c/ aragón, 272 8º 8ª
tel. 215 69 52
barcelona-7

CONTABILIDAD 64

CONTABILIDAD 64

- Programa adaptado al Plan General Contable Español
- Cuentas de Mayor (3 dígitos) y de detalle (6 dígitos)
- 300 cuentas y 3000 apuntes por diskette
- Programa en cartucho y diskette simultáneamente
- Equipo: ordenador COMMODORE 64 ó SX-64, unidad de disco VC-1541, impresora VC-1525 ó MPS-801 y monitor ó T.V.

MENU

1 ASIENTOS, DIARIO y ...
1 Entrada
2 ...

MENU

- completo MANUAL de AN GENERAL
- 1 ASIENTOS, DIARIO Y CONSULTAS
1 Entrada de asientos
2 Diario
3 Consulta de cuentas
- 2 EXTRACTOS Y SUMAS Y SALDOS
1 Extractos de cuentas
2 Balance de Sumas y Saldos
- 3 SITUACION Y CIERRE
1 Balance de Situación
2 Cuenta de Explotación
3 Diario de Cierre
- 4 MANTENIMIENTO DE FICHEROS
1 Parámetros
2 Cuentas (altas-bajas-modificaciones)
3 Asientos (bajas-modificaciones)
- 5 LISTADO DE FICHEROS
1 Parámetros
2 Cuentas
3 Asientos
- 6 UTILITARIOS
1 Copia de disco
2 Borrado de periodo
3 Vaciado de asientos
4 Regeneración ficheros
5 Regeneración/verificación disco
6 programación balance

Se incluye un completo MANUAL DE UTILIZACION asi como un resumen del PLAN GENERAL CONTABLE ESPAÑOL y un capítulo con NOCIONES FUNDAMENTALES DE CONTABILIDAD para hacer posible la utilización de este programa a personas no expertas en contabilidad.

P. V. P. 24550..

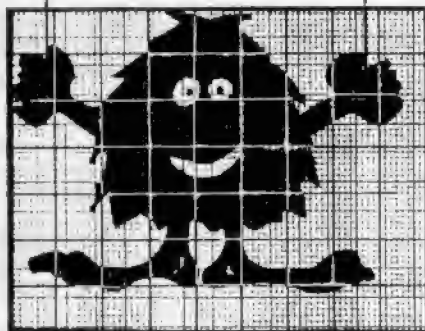
EMPRESA : CASA DE SOFTWARE, S.A.	
PASIVO	
CAPITAL Y RESERVAS	
100 CAPITAL	
DEUDAS CORTO	
400 PROVEEDORES	
400 EFECTOS C/ALES A PP	
TOTAL PASIVO	
EMPRESA : CASA DE SOFT	
ACTIVO	
INMOVILIZADO	
MATERIAL	
200 MAQUINARIA	
200 OTRO INMOB	
EXISTENCIAS	
300 COMERCIA	
310 PRODUCTO	
350 MATERIA	
DEUDORES	
400 CLIENTE	
OTAS. FINANCI	
570 C/AJA	
572 BANC	
RESULTADOS	
TOTAL ACT	

[illegible]

FECHA	16-02-84	HORA	5
	DEBE	HABER	
	3341000.00	3041000.00	
	0.00		
INRADIOS	0.00		
Y Y REG.	0.00		
		450000.00	
IS		20000.00	
Y FLETES		1000.00	
OFICINA		20000.00	
	50000.00		
		50000.00	
	491000.00		
		441000.00	
GRANUJINES	441000.00		
	450000.00		

DISC 1 FROM 18-02-84				MOJG
CONTRACT	DOK	INPGPTE		
PROG. 1. (1000)	D	1000000,00		
PROG. 2. (1000)	M	1000000,00		
PROG. 3. (1000)	M	1000000,00		
PROG. 4. (1000)	D	1000000,00		
PROG. 5. (1000)	D	1000000,00		
PROG. 6. (1000)	D	1000000,00		
PROG. 7. (1000)	D	1000000,00		
PROG. 8. (1000)	D	1000000,00		
PROG. 9. (1000)	D	1000000,00		
PROG. 10. (1000)	D	1000000,00		
PROG. 11. (1000)	D	1000000,00		
PROG. 12. (1000)	D	1000000,00		
PROG. 13. (1000)	D	1000000,00		
PROG. 14. (1000)	D	1000000,00		
PROG. 15. (1000)	D	1000000,00		
PROG. 16. (1000)	D	1000000,00		
PROG. 17. (1000)	D	1000000,00		
PROG. 18. (1000)	D	1000000,00		
PROG. 19. (1000)	D	1000000,00		
PROG. 20. (1000)	D	1000000,00		
PROG. 21. (1000)	D	1000000,00		
PROG. 22. (1000)	D	1000000,00		
PROG. 23. (1000)	D	1000000,00		
PROG. 24. (1000)	D	1000000,00		
PROG. 25. (1000)	D	1000000,00		
PROG. 26. (1000)	D	1000000,00		
PROG. 27. (1000)	D	1000000,00		
PROG. 28. (1000)	D	1000000,00		
PROG. 29. (1000)	D	1000000,00		
PROG. 30. (1000)	D	1000000,00		
PROG. 31. (1000)	D	1000000,00		
PROG. 32. (1000)	D	1000000,00		
PROG. 33. (1000)	D	1000000,00		
PROG. 34. (1000)	D	1000000,00		
PROG. 35. (1000)	D	1000000,00		
PROG. 36. (1000)	D	1000000,00		
PROG. 37. (1000)	D	1000000,00		
PROG. 38. (1000)	D	1000000,00		
PROG. 39. (1000)	D	1000000,00		
PROG. 40. (1000)	D	1000000,00		
PROG. 41. (1000)	D	1000000,00		
PROG. 42. (1000)	D	1000000,00		
PROG. 43. (1000)	D	1000000,00		
PROG. 44. (1000)	D	1000000,00		
PROG. 45. (1000)	D	1000000,00		
PROG. 46. (1000)	D	1000000,00		
PROG. 47. (1000)	D	1000000,00		
PROG. 48. (1000)	D	1000000,00		
PROG. 49. (1000)	D	1000000,00		
PROG. 50. (1000)	D	1000000,00		
PROG. 51. (1000)	D	1000000,00		
PROG. 52. (1000)	D	1000000,00		
PROG. 53. (1000)	D	1000000,00		
PROG. 54. (1000)	D	1000000,00		
PROG. 55. (1000)	D	1000000,00		
PROG. 56. (1000)	D	1000000,00		
PROG. 57. (1000)	D	1000000,00		
PROG. 58. (1000)	D	1000000,00		
PROG. 59. (1000)	D	1000000,00		
PROG. 60. (1000)	D	1000000,00		
PROG. 61. (1000)	D	1000000,00		
PROG. 62. (1000)	D	1000000,00		
PROG. 63. (1000)	D	1000000,00		
PROG. 64. (1000)	D	1000000,00		
PROG. 65. (1000)	D	1000000,00		
PROG. 66. (1000)	D	1000000,00		
PROG. 67. (1000)	D	1000000,00		
PROG. 68. (1000)	D	1000000,00		
PROG. 69. (1000)	D	1000000,00		
PROG. 70. (1000)	D	1000000,00		
PROG. 71. (1000)	D	1000000,00		
PROG. 72. (1000)	D	1000000,00		
PROG. 73. (1000)	D	1000000,00		
PROG. 74. (1000)	D	1000000,00		
PROG. 75. (1000)	D	1000000,00		
PROG. 76. (1000)	D	1000000,00		
PROG. 77. (1000)	D	1000000,00		
PROG. 78. (1000)	D	1000000,00		
PROG. 79. (1000)	D	1000000,00		
PROG. 80. (1000)	D	1000000,00		
PROG. 81. (1000)	D	1000000,00		
PROG. 82. (1000)	D	1000000,00		
PROG. 83. (1000)	D	1000000,00		
PROG. 84. (1000)	D	1000000,00		
PROG. 85. (1000)	D	1000000,00		
PROG. 86. (1000)	D	1000000,00		
PROG. 87. (1000)	D	1000000,00		
PROG. 88. (1000)	D	1000000,00		
PROG. 89. (1000)	D	1000000,00		
PROG. 90. (1000)	D	1000000,00		
PROG. 91. (1000)	D	1000000,00		
PROG. 92. (1000)	D	1000000,00		
PROG. 93. (1000)	D	1000000,00		
PROG. 94. (1000)	D	1000000,00		
PROG. 95. (1000)	D	1000000,00		
PROG. 96. (1000)	D	1000000,00		
PROG. 97. (1000)	D	1000000,00		
PROG. 98. (1000)	D	1000000,00		
PROG. 99. (1000)	D	1000000,00		
PROG. 100. (1000)	D	1000000,00		

¡Y YO TAMBIÉN
CUENTO!



RINCON
DEL 700



Software para el 700 (II)

El aspecto más importante del 700 en lo que a software se refiere es MEC/DOS, sistema operativo desarrollado por el Departamento de Software de Microelectrónica y Control, S. A. que incorpora 52 comandos más al Basic de Commodore para tratamiento de Ficheros (Relativos, Tablas y de Claves —Indexados—), gestión de Pantallas, Múltiple Precisión, Empaquetado y Desempaquetado de Campos, etc... En la VENTANA CBM de este número se describe, en líneas generales lo que es MEC/DOS y lo que representa para el 700. Por ahora, seguiremos aquí hablando de software para el 700, aunque suponiendo siempre que este soft se confecciona con MEC/DOS.

La potencia que puede tener el 700 con MEC/DOS y un disco Winchester es sumamente considerable. Con diskettes, esta potencia está condicionada al modo

de emplearlos (tipos y frecuencia de accesos), y a la capacidad de los mismos. No obstante, la disponibilidad de los ficheros de tipo Tabla (MEC/DOS) permite paliar esta posible desventaja que puedan ocasionar los diskettes en algunas ocasiones, operando en RAM los pequeños ficheros que pueden entorpecer el acceso a los grandes.

El secreto de una buena programación está en trabajar con el número de ficheros más apropiado en cada caso, y en el correcto diseño de los mismos. En los ficheros no pueden haber campos inútiles que entorpezcan el uso de los que realmente sí son necesarios. En los ordenadores, y más concretamente en los micros, el análisis debe esforzarse al máximo en su cometido para evitar que la máquina se convierta en un "muerto" que cause más problemas que ventajas a sus usuarios.

Todos sabemos lo importante que es el análisis en el desarrollo de las aplicaciones. El análisis es el punto intermedio resultante de lo que el usuario quiere que haga su ordenador (a veces verdaderos sueños imposibles) y lo que realmente puede hacerse en él. El analista es el intérprete entre el lenguaje profano en informática del usuario y la jerga propia de los programadores.

Pero más importante aún que el análisis es la filosofía general de trabajo de un analista, programador, o casa de software. En efecto, en lugar de estudiar cada aplicación por separado, es necesario tener unas normas u objetivos generales para que todos los paquetes tengan una estructura similar; ello repercute en claridad de análisis, rapidez de programación, y otras enormes ventajas que más adelante veremos.

En este artículo voy a comentar a nivel de análisis un paquete de gran divulgación. El tema es ya muy conocido, pero

seguro que destaco algún aspecto interesante útil a alguien, o sugiero alguna idea nueva a algún analista.

El programa que casi todo usuario de ordenador tiene instalado es el de Contabilidad. Es el paquete más estandarizado que pueda haber (junto con el procesador de textos), pero también es cierto que es el que más variantes tiene. Básicamente, un programa contable debería hacer las siguientes tareas:

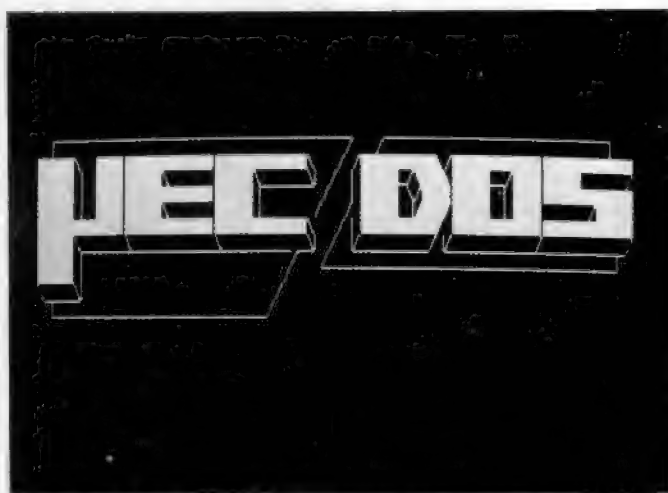
- Mantenimiento de Ficheros (Altas, Bajas y Modificaciones).
- Entrada de Apuntes y Corrección de los mismos.
- Listado Mayor.
- Listado Diario.
- Balances (Sumas y Saldos, Situación y Explotación).
- Cierre de Ejercicio.

Estos son los puntos básicos. Si un programa de Contabilidad no realiza alguno de ellos, es que ya ni siquiera es operativo.

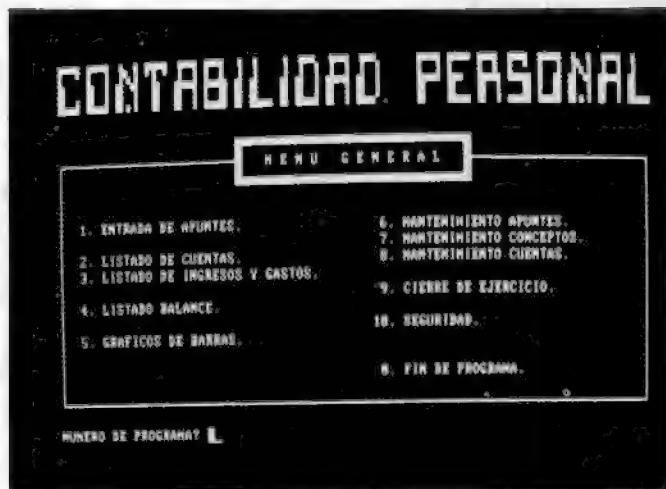
A partir de aquí, los "softwareros" pueden añadirle todos los adornos que quieran para dar más importancia al paquete:

- Cartera de Efectos a Cobrar.
- Cartera de Efectos a Pagar.
- Control presupuestario (mensual o total).
- Estadística de Ratios.
- Listados alfabéticos de cualquier fichero.
- Confección de Balances atrasados (por mes).
- Programación manual de los Balances de Situación y Explotación, pudiendo operar con varios formatos.
- Diversos controles de cuadre.
- Captación de apuntes generados por otros programas.
- ...y un largo etcétera.

El problema se presenta cuando una empresa necesita un listado que su pro-



Logotipo de MEC/DOS tal como aparece en la pantalla de un CBM-B-700.
(Diseño: J. Sastre. Foto: P. Masats).



Aspecto de la pantalla "Menú General" de un módulo de contabilidad personal realizado con MEC/DOS en un CBM-B-700. (Diseño: J. Sastre. Foto: P. Masats).

grama no es capaz de hacer, y cuando una empresa tiene que soportar todas las funciones extras que su programa hace, pero que en realidad no le hacen falta. En el primer caso el ordenador está infrautilizado (mal programado) o es demasiado pequeño; mientras que en el segundo está mal instalado: si se suprimieran del programa todos los procesos que el usuario no utiliza, seguro que los tiempos de ejecución bajarían bastante y la capacidad de almacenamiento para apuntes y cuentas se incrementaría considerablemente.

Este problema tiene varias soluciones, algunas más factibles que otras. Las casas de software podrían tener una amplia gama de programas de Contabilidad, cada uno enfocado a un tipo de empresa diferente. Ello exigiría un notable esfuerzo de la misma en el mantenimiento de tales programas. Pueden tener un programa-base al que implementarían todas las opciones que el usuario deseara, como si hicieran los programas de Contabilidad "a medida". El inconveniente es el mismo que tienen todas las aplicaciones "a medida": el tiempo de análisis, de programación, de pruebas, de depuración, etcétera, implica un alto coste del programa. Además que se encontrarían con que cada cliente tiene un programa diferente, a saber cuál es el mejor. Una solución más factible es tener el programa repartido en módulos interactivos, que el usuario elegiría a su gusto. Este es el sistema más deseable, pero el más difícil de conseguir. Podría estar compuesto de:

- Programa base (puntos mencionados al principio).
- Módulo de Cartera de Efectos.
- Módulo de Presupuestos.
- Módulo de Ratios.
- Módulo de Estadística Mensual.
- ...etc.

Existe también un caso, sobre todo en las personas que trabajan como profesionales liberales, en que ya el módulo base de contabilidad le viene grande. Fiscalmente no están obligados a llevar una contabilidad oficial, por lo que a ellos les bastaría controlar sus ingresos y gastos particulares. Tendrían suficiente con un programa que, además del mantenimiento de ficheros y entrada de apuntes, les confeccionara un detalle de ingresos y gastos,

el listado de cuenta corriente del banco, y un balance que les informara de los beneficios. El escaso número de cuentas contables (básicamente cuentas de ingresos, gastos, caja y bancos) y de apuntes le redundaría en una mayor velocidad de ejecución y en la posibilidad de completar sus listados con diagramas de barras, estadística mensual, etc...

Actualmente, en nuestro país, por muy baratos que sean los microordenadores, no llegan a justificar su precio si se adquieren para llevar únicamente esta mini-contabilidad. Hay ordenadores personales, incluso familiares, capaces de gestionarla, introduciendo los apuntes después del combate intergaláctico y antes de la partida de comecocos. Pero un profesional de este tipo tiene otros trabajos que también son mecanizables: el abogado necesita un procesador de textos, el arquitecto un cálculo de estructuras, el constructor un seguimiento de obras, el que esté sujeto a la Estimación Objetiva Singular una relación de facturas emitidas, etc... A veces un ordenador es más útil cuando realiza las pequeñas tareas de cada uno, que mecanizando todo un complejo y voluminoso proceso.

Normalmente, el ordenador se compra para que realice el mayor número de tareas posible, con el mínimo presupuesto de maquinaria; es decir, mucho soft con poco hard. Si a eso le añadimos que, por definición popular infundada, el soft tiene que ser más barato que el hard, los distribuidores de ordenadores se las ven negras para confeccionar un soft barato que a la vez sea potente, cosas casi siempre incompatibles. Han recurrido a los estándares anunciando grandes programas de contabilidad, grandes programas de facturación y grandes programas de nóminas a un precio realmente barato. Han preferido hacer un gran paquete que lo pueda "hacer todo", antes que confeccionar cada una de las aplicaciones a medida. Ello repercute en los que realmente no necesitan tantos procesos y listados para mecanizar su empresa.

Hemos llegado al problema inicial planteado en este artículo: el programa estándar que no se adapta al usuario. Hasta que no arraigue entre el gran conjunto de consumidores de informática la idea de

que la programación puede resultar más cara que el propio ordenador, el problema seguirá existiendo. Cuando ello ocurra, estos grandes paquetes podrán dividirse en otros más pequeños que se podrán modularizar entre sí. Por ejemplo, la gestión comercial de una empresa estará compuesta por:

- Módulo de Contabilidad (a su vez también modularizado).
- Módulo de Control de Almacén.
- Módulo de Control de Clientes.
- Módulo de Control de Proveedores.
- Módulo de Facturación (facturas, recibos y remesas).
- Y todos los módulos que hagan falta.

De esta manera, pueden haber muchos módulos sobre un mismo tema, por ejemplo el Control de Clientes: algunos controlarán las compras totales que hace cada cliente, otros lo controlarán mensualmente, habrán diferentes modalidades de pago pre-programadas, algunos módulos incorporarán el cálculo de rappels, otros tendrán estadística de compras por unidad, litro, kilo, etc... La elección apropiada de cada uno de los módulos que componen la programación permitirá obtener una aplicación casi a medida. Cuantos más módulos haya por elegir, más eficaz será la mecanización total.

Por ahora, los costes que implican confeccionar cada uno de estos módulos, que han de ser compatibles con los demás, aparte de poder funcionar por sí solos, es demasiado alto en el campo de los microordenadores. Estaremos más cerca de ello si empezamos a confeccionar pequeños paquetes, réplica de los grandes, que puedan comunicarse con algún otro pequeño paquete. El coste de la mecanización total de una oficina, a base de estos pequeños paquetes, puede resultar más caro, pero también más ventajoso al usuario, que dispondrá de programas a medida que no tienen etapa de depuración, pues están ya confeccionados. En realidad, cada uno de estos pequeños paquetes puede ser un módulo de una programación mayor.

Todo lo dicho no afecta únicamente a la serie 700 de Commodore, ni a los ordenadores Commodore en conjunto, pero ya estamos en lo de siempre: quien pega primero pega dos veces.

EL RINCON DEL VICCOLAGE



Los capítulos previos fueron publicados con anterioridad en **COMMODORE CLUB**. A aquellos lectores que los deseen, les rogamos que nos lo comuniquen para poder enviarles copia de estos capítulos.

El RS-232-C (II)

El RS-232-C en los ordenadores Commodore

Las últimas generaciones de ordenadores **COMMODORE**, (**VIC 20**, **COMMODORE 64**, **CBM 700**), incorporan de alguna manera el hardware necesario para efectuar la comunicación con otros periféricos u ordenadores mediante un canal **RS-232-C**. El **VIC 20** y el **COMMODORE 64** requieren un cartucho de interface, el **VIC 1011A**, que se conecta en el port del usuario de cualquiera de ellos. El **CBM 700** incorpora de origen un port de comunicación serie **RS-232-C**.

El **RS-232-C** del **COMMODORE 64** y del **VIC 20** está dispuesto en el formato estándar de esta interface pero los niveles no son los adecuados del **RS-232-C**. El interface de **COMMODORE (VIC 1011A)** efectúa esta adaptación.

El acceso y programación del interface puede efectuarse desde **BASIC** o mediante ciertas rutinas del **KERNAL**.

El software para manejar la interface desde **BASIC** utiliza las siguientes instrucciones: **OPEN**, **CLOSE**, **CMD**, **INPUT#**, **GET#**, **PRINT#**, y la variable reservada **ST**.

Las instrucciones **INPUT#** y **GET#** toman los datos desde el buffer receptor y **PRINT#** y **CMD** colocan los datos en el buffer emisor.

Apertura de un canal RS-232-C

Sólo se puede abrir un canal de este tipo a la vez. Una segunda instrucción **OPEN** causa el reajuste de los punteros del buffer. Todos los caracteres de cualquiera de los dos buffers se perderán.

Se pueden colocar hasta 4 caracteres en el campo del nombre del fichero. Los dos primeros son caracteres de control y de comando. Los otros dos están reservados para sistemas futuros. La velocidad de transmisión, paridad y otras opciones se pueden seleccionar usando esta característica. Si se utiliza un carácter de control ilegal o que no está definido, el sistema envía los datos a una velocidad menor de 50 baudios.

Sintaxis en basic

La sintaxis utilizada para la apertura de un canal **RS-232-C** es la siguiente:

OPEN lfn,2,sa,
—REGISTRO DE CONTROL—
—REGISTRO DE COMANDO—

Logic filename —lfn.— El número de fichero lógico puede ser cualquier número entre 1 y 255. Si se utiliza un número superior a 128 se alimentará una línea después de cada retorno de carro.

Secondary address —sa.— La dirección secundaria en el VIC 20 y el COMMODORE 64 es recomendable que sea siempre un cero. En el CBM 700 existen varias opciones:

sa=0 - NO ACTUA
sa=1 - SOLO TRANSMISION
sa=2 - SOLO RECEPCION
sa=3 - TRANSMISION/RECEPCION
sa mayor que 128 - HABILITA LA CONVERSION CBM/ASCII

Registro de control

Es un carácter de un solo byte que contiene la información de velocidad de transmisión/recepción, longitud de palabra, número de bits de stop. Vea la figura 1.

Registro de comando

Es un carácter de un solo byte que define otros parámetros del terminal como son: Opciones de paridad, half/full duplex, modo de Handshake. Ver figura 2.

En un programa BASIC, la apertura de un canal RS-232-C debe hacerse antes de definir cualquier variable o tabla ya que automáticamente, al abrir un canal RS-232-C se efectúa un CLR. También se ha de tener en cuenta que son necesarios 512 bytes libres como mínimo antes de la apertura de un canal RS-232-C.

Para que quede claro el concepto de REGISTRO DE CONTROL y REGISTRO DE COMANDO pondremos un ejemplo. Supongamos que deseamos programar la interface como 7 bit de longitud de palabra, 1 bit de stop, 300 bauds de velocidad de recepción/transmisión, paridad impar recepción/transmisión, full duplex y 3 líneas de comunicación.

El valor binario del registro de comando es 00100000 y su valor en decimal es 32.

Por lo tanto utilizando la sintaxis en BASIC debemos poner:

OPEN 2,2,0,chr\$(38)+chr\$(32)

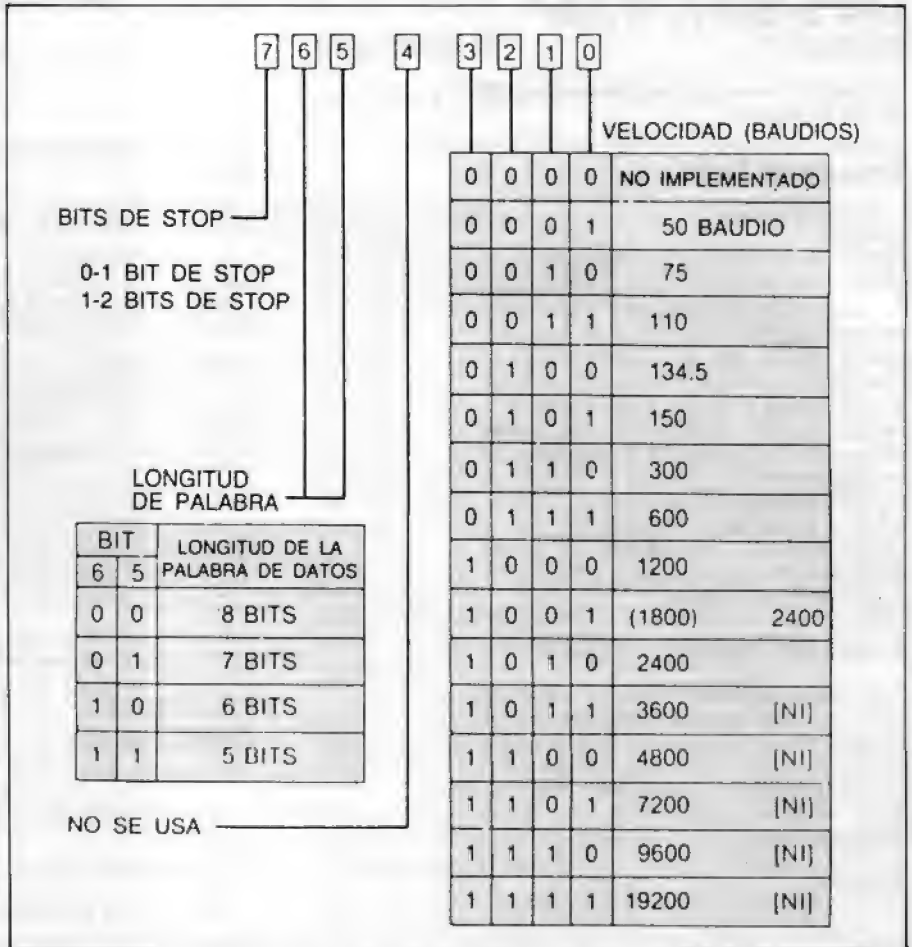


Figura 1. Estructura del Registro de Control (VIC-20 y C-64)

Recepción de datos desde un canal RS-232-C

En la recepción de datos de un canal RS-232-C, el buffer del VIC 20 y del COMMODORE 64 pueden almacenar hasta 255 caracteres antes de llenarse por completo. En el caso de que se intente almacenar más caracteres de los indicados se produce un OVERFLOW que es indicado en el bit correspondiente de la palabra de estado.

Si ocurre un desbordamiento de la capacidad del buffer, todos los caracteres recibidos desde que el buffer estaba completo se perderán.

Si desea recibir datos desde un periférico RS-232-C de gran velocidad deberá utilizar un programa en código máquina, ya que el BASIC es demasiado lento y probablemente perdería caracteres.

Sintaxis en Basic

La sintaxis recomendada es:
GET lfn, —VARIABLE DE CADENA—
Si la palabra es menor a 8 bits, todos los bits no utilizados contendrán el valor cero.

Si un GET no encuentra datos en el buffer, devuelve una cadena vacía.

Si se usa INPUT, el sistema espera hasta que encuentra una cadena no vacía seguida de un retorno de carro. Por esto, si las señales CLEAR TO SEND o DATA SET READY desaparecen durante la ejecución del INPUT, la ejecución del programa se interrumpe. Por esto no se recomienda el uso de INPUT.

Bit núm.	7	6	5	4	3	2	1	0
1 bit stop.....	0							
7 bit datos.....		0	1					
No se usa.....				0				
300 bauds.....					0	1	1	0
El valor binario del registro de control es 00100110 y su valor en decimal es 38.								
Bit núm.	7	6	5	4	3	2	1	0
Paridad impar R/T.....	0	0	1					
Full duplex.....				0				
No se usa.....					0	0	0	
3 líneas.....								0



Envío de datos por el canal RS-232-C

El buffer de salida tiene una capacidad de 255 caracteres.

Sintaxis en Basic

Se puede utilizar:

CMDIfn
PRINT Ifn, —LISTA DE VARIABLES—

No existe un retorno de carro en canal de salida. Esto significa que una impresora normal RS-232-C puede no imprimir bien a menos que se implemente algún tipo de HOLD-OFF (se pida al ordenador que espere), o exista algún buffer en la impresora. El HOLD-OFF se puede improvisar fácilmente en el programa. Si se trabaja en modo X-líneas el VIC 20 o el COMMODORE 64 efectúan el HOLD-OFF a partir de la señal CTS. En el COMMODORE 64 y el VIC 20 se han implementado las líneas RTS, CTS, DCD, como terminal de datos.

Cierre del canal RS-232-C

El cierre del canal RS-232-C descarta todos los datos presentes en los buffers en el momento de la ejecución (hayan sido o no transmitidos o impresos), detiene todas las transmisiones y recepciones del RS-232-C, ajusta la línea RTS y las líneas altas (Sout) de datos transmitidos, y destruye el contenido de los dos buffers del RS-232-C.

Sintaxis en Basic

CLOSEIfn

Asegúrese de que se han transmitido todos los datos antes de cerrar el canal serie. Una forma de verificar esto desde BASIC puede ser:

```
100SS=ST:IF(SS=0 OR SS=8) THEN 100
110CLOSEIfn
```

Variable ST o byte de estado. (Ver fig. 3)

La variable reservada o variable de estado ST de longitud 1 byte, contiene

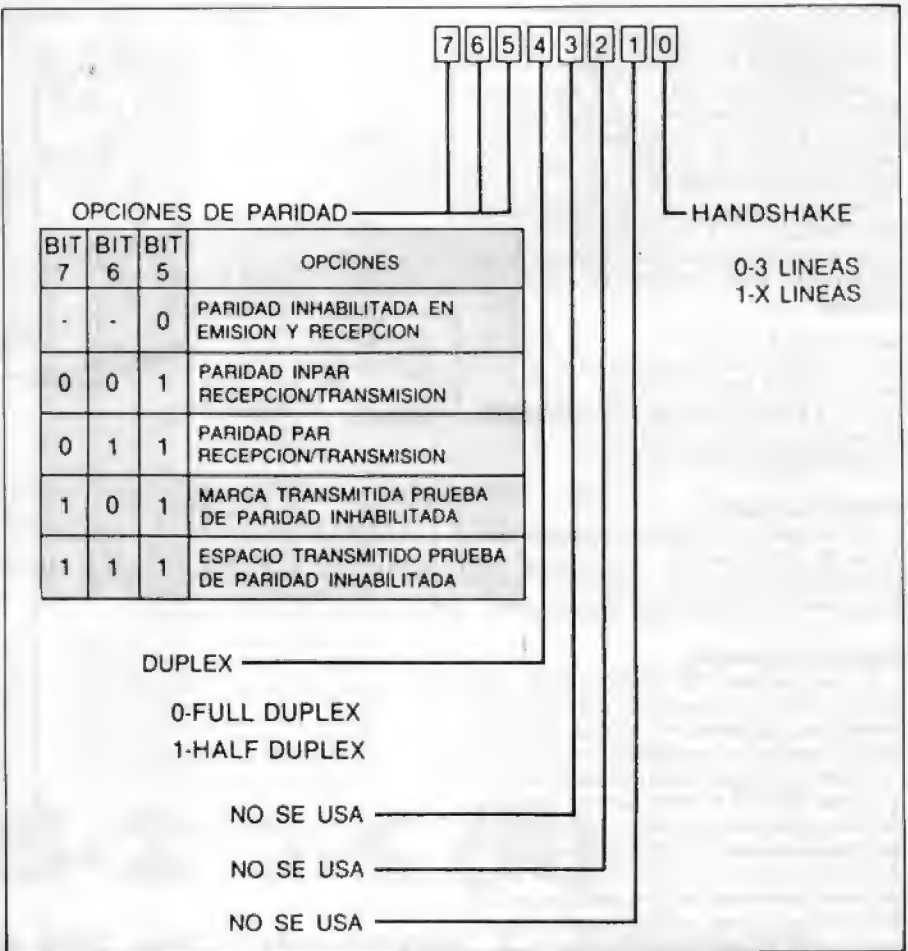


Figura 2.
Estructura del Registro de Comando (VIC-20 y C-64).

toda la información necesaria para la validación de cualquier dato enviado o leído por el canal serie. Si los bits están a cero no se detecta error. La lectura de la palabra de estado provoca su borrado, y por tanto en los casos en que sean necesarias varias lecturas de la misma, se procederá a su asignación a otra variable. La variable de estado puede ser leída sólo cuando el canal serie ha sido el último en utilizarse en una operación de E/S.

Detalles de los conectores del canal RS-232-C

SEÑAL	VIC 20/C 64	VIC 1011/CBM 700	SENTIDO
Gnd	A,N	1,7	
RxD	B,C	3	IN
TxD	M	2	OUT
RTS	D	4	OUT
DTR	E	20	OUT
DCD	H	8	IN
DSR	L	6	IN
CTS	K	5	IN

Figura 3.
Registro de Estado (ST) del RS-232.

[7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]	
1	1	1	1	1	1	1	1	PARITY ERROR BIT
1	1	1	1	1	1	1	1	FRAMING ERROR BIT
1	1	1	1	1	1	1	1	RECEIVER BUFFER OVERRUN BIT
1	1	1	1	1	1	1	1	RECEIVER BUFFER—EMPTY
1	1	1	1	1	1	1	1	(USE TO TEST AFTER A GET#)
1	1	1	1	1	1	1	1	CTS SIGNAL MISSING BIT
1	1	1	1	1	1	1	1	UNUSED BIT
1	1	1	1	1	1	1	1	DSR SIGNAL MISSING BIT
1	1	1	1	1	1	1	1	BREAK DETECTED BIT



Una Excursión en Basic Más allá del Manual

Capítulo II

Por Jeffrey Mills

En este capítulo vamos a descubrir cómo los valores se asignan a las variables — esto es un truco que cada programador debe tener en su repertorio.

En la Primera Parte de esta serie de artículos, empezamos a desarrollar un programa capaz de catalogar y listar los programas según un número de cinta.

Hablamos de los comandos preparatorios New, CLR y List. También hablamos de la numeración de las REM y las líneas. Hasta ahora, nuestro programa tiene el siguiente formato:

```
10 REM CATALOGO PROGRAMA/CINTA
20 REM ESCRITO POR: tu nombre
30 PRINT "(Shift-CLR/HOME)"
40 PRINT "CATALOGO CINTA"
50 PRINT
60 PRINT "101", "JUEGO 1"
70 PRINT "101", "JUEGO 2"
```

Ahora volvamos atrás para añadir las cabeceras de las columnas a nuestra lista. Utilizaremos la sentencia PRINT con comas. Se tecla:

```
52 PRINT "CINTA", "PROGRAMA"
54 PRINT "NO.", "NOMBRE"
56 PRINT
```

Hay que observar la forma en que las líneas se insertan entre los existentes números de línea. Esto constituye un buen ejemplo del motivo por el cual las líneas se numeran de diez en diez al comenzar un listado.

En vez de utilizar una distinta sentencia PRINT para cada programa de la lista, existe un método más sencillo. Sin embargo, antes de empezar a hablar de esto, hace falta saber un par de cosas más.

Más Terminología

1) Variable.—Una variable es una pequeña porción de la memoria del ordenador, a la cual se le asigna un valor, u otro contenido que podría cambiar, o variar, en el curso de un programa. Las variables podrían ser consideradas como pequeñas cajas o cajones de ficheros creados en la memoria mediante una sentencia del programa.

Existen varios tipos de variables. Para nuestros fines actuales, sólo hablaremos de las variables numéricas o de serie (letra o carácter).

A una variable en un programa se le puede asignar cualquier nombre dentro de las siguientes normas:

- Debe de empezar con una letra del alfabeto (A-Z), pero se puede utilizar tanto una letra como un número (0-9) en la segunda posición del nombre.

- Las dos primeras letras del nombre deben ser distintas de las dos primeras letras de cualquier otro nombre elegido. Estas dos letras son las que utiliza el ordenador para distinguir un cajón de fichero de otro. Por ejemplo, para el ordenador, BOY y BOAT son iguales, ya que sólo se miran las dos primeras letras.

- No se puede utilizar cualquier nombre de variable que empiece con las mismas dos letras que una Palabra Clave en Basic. (El Apéndice D del manual.)

Si la variable sólo va a contener números, se utiliza una variable numérica. Si va a contener caracteres (letras y símbolos), se utiliza una variable de serie.

Las normas para los nombres variables son las mismas que para las variables numéricas y la serie. Se le indica al ordenador que la variable va a contener letras y símbolos mediante la adición de un signo de dólar (\$) al final del nombre.

Normalmente es mejor que los nombres variables sean lo más cortos posible, dado que cada carácter ocupa espacio en la memoria del ordenador. Sin embargo, si se llama a una variable que almacena un nombre NAME\$, resultará fácil repasar el programa en el futuro para saber lo que representa dicha variable.

2) Sentencia de asignación.—Ahora que se sabe que el ordenador puede almacenar información en los cajones de ficheros llamados variables, hay que aprender la forma de introducir los valores en dichas variables. Esto se hace mediante una sentencia assignment (Asignación).

Supongamos que una variable se llama "A" y tiene que asumir el valor 5. Es suficiente teclear A=5. (También se puede teclear LET A=5, pero el comando LET se utiliza pocas veces, dado que el Commodore supone que se entiende LET si se teclea A=algún valor X.

Si se quiere cambiar el valor de la variable A a otra cosa, digamos 12, al teclear A=12, se borrará el 5, y el 12 se colocará en su lugar en el cajón de fichero A.

Leyendo los Datos

3) Read/Data.—El uso de los dos comandos Read y Data facilita la asignación de valores a las variables. Una explicación sobre estas dos sentencias Basic se presenta en las páginas 92 a 94 del manual.

La sentencia Data no es ejecutable. No es un verbo. El ordenador no hace nada cuando ve esta palabra en un programa.

Una sentencia Data simplemente almacena los números y palabras que serán utilizados por el programa. Se listan en el orden en que se utilizarán, y se separan mediante el uso de comas. A diferencia de la sentencia PRINT, estas comas no afectan en absoluto la salida de información en la sentencia Data. Su función es la de identificar la separación entre dos unidades de datos.

Cuando un programa se ejecuta, el ordenador recoge toda la información contenida en una sentencia Data y la almacena en una posición especial. De esta forma, la información puede ser recuperada en el momento en que la pide el programa. El usuario le indica al ordenador que recupere información de una sentencia Data, mediante el comando Read.

El comando Read recoge el siguiente valor de la lista, lo coloca en la variable especificada, y desplaza un puntero a la siguiente unidad de información de la lista. El puntero le indica a la siguiente sentencia Read dónde tiene que empezar.

En general, una vez que una unidad de información haya sido leída, no se vuelve a tener acceso a ella. El comando Read repasa la lista, leyendo toda la información, hasta el final de ésta. (Si el usuario intenta leer más información que la contenida en la lista de sentencia Data, el programa terminará con un mensaje de error de Out of Data [Faltan Datos]).

A continuación, se presenta una típica sentencia Data:

```
90 DATA 2,42,56,13
```

A continuación, se presenta una típica sentencia Read:

```
100 READ A
```

Mediante el uso del comando Read, el usuario le indica al ordenador que lea la siguiente unidad de información de la lista de sentencias Data y que coloque lo que lea en la variable especificada en la sentencia Read.

Cuando se ejecuta la línea mencionada anteriormente, A contendrá el número 2. Si la siguiente línea fuese 110 READ B, la variable B contendría el valor 42, y A seguiría conteniendo el valor 2.

Las sentencias Data también pueden contener datos alfabéticos. Por ejemplo:

```
90 DATA CAT, DOG, BIRD
```

Ahora la sentencia Read tendrá que especificar una variable de serie. Por ejemplo:

```
100 READ A$
```

Cuando se ejecuta esta sentencia, la variable llamada A\$ (pronunciado A-string) contendrá la palabra "cat". Si la línea 110 se cambiara ahora a 110 READ B\$, B\$ contendría la palabra "dog", y A\$ seguiría con su inquilino felino.

También se pueden mezclar datos numéricos y de serie (números y palabras) dentro de una sentencia Data. Por ejemplo:

```
90 DATA 1, GAME 1, 1GAME 2
```

Ahora las sentencias Read serían de la siguiente forma:

```
100 READ A
```

```
110 READ A$
```

```
120 READ B
```

```
130 READ B$
```

Después de ejecutarse estas sentencias, A contendrá el valor 1, A\$ contendrá el "string" de caracteres GAME 1, B contendrá el valor 1 y B\$ el "string" de caracteres GAME 2.

De la misma forma en que se utilizan las comas para separar las unidades de información dentro de una sentencia Data, se puede especificar que se lea más de una unidad de información en una sentencia Read. Por ejemplo:

```
100 READ A,A$,B,B$
```

Esta sentencia realiza lo mismo que las cuatro líneas anteriores.

Volviendo al Programa...

Dado que ya sabemos asignar valores a las variables, podemos imprimir el listado usando solamente una sentencia Print (después de las cabeceras). Sin embargo, hay otro concepto que debemos de tocar para poder realizar estas tareas de una forma efi-

caz. Este consiste en el bucle GOTO, del cual hablaremos con más detalle en otro capítulo.

De momento, es suficiente recordar que la sentencia GOTO le envía al ordenador a un número de líneas determinado dentro del programa. Si esto se hace al final de una serie de líneas, el ordenador vuelve a realizar el mismo juego de instrucciones una y otra vez, hasta que ocurre algo que produce un cambio. (La sentencia GOTO se explica en las páginas 30 y 123 del manual).

Vamos a sustituir las líneas 60 y 70 en nuestro programa Catalog por una sentencia Read y una sentencia Print. Se teclean las siguientes líneas:

```
60 READ N,P$
```

```
70 PRINT N,P$
```

Ahora se supone que nos hace falta una sentencia Data para acompañar la sentencia Read. Para que no resulte demasiado complicado, las sentencias Data tendrán asignados unos números de línea lo suficientemente altos como para permitir la inserción de otras líneas. Vamos a empezar con el número de líneas 9000. Se teclea:

```
9000 DATA 101,GAME 1,GAME 2
```

```
9010 DATA 102,GAME 3,GAME 4
```

```
9020 DATA 103,GAME 5,GAME 6
```

Las unidades de información contenidas en cada una de estas sentencias son un número de cinta, un nombre de juego, un número de cinta y un hombre de juego. Si se ejecuta el programa ahora, sólo se imprimirá el primer número de cinta y el primer nombre de juego. Aquí es donde entra en juego el bucle GOTO.

El programa entra en un bucle si se teclea 80 GOTO 60. Volverá a ejecutar las líneas 60 y 70, pero cuando termina la línea 80, regresará a la línea 60. Volverá a ejecutar las líneas 60 y 70 otra vez... y otra... y otra...

Cuando el programa haya leído toda la información contenida en la lista de sentencias Data, el programa termina imprimiendo lo siguiente en pantalla:

```
?OUT OF DATA ERROR EN 60
```

Existe una lista de todos los mensajes de error del Commodore en la página 150 y 151 del manual. Es normal que no signifiquen gran cosa para el principiante en estos momentos. Seguramente nos tropezaremos con muchos de ellos en nuestra pantalla a medida que vayamos escribiendo programas en Basic.

El Ejemplo 1 demuestra la salida del programa Catalog tal y como lo hemos desarrollado hasta ahora.

En el siguiente artículo hablaré de los métodos empleados para que el programa no termine en error, además de las formas de controlar la formación de bucles en el programa dentro del bucle GOTO. ■

Ficheros en disco (VI)

Indexados Secuenciales

Por Manuel AMADO

a) Introducción:

Este artículo pretende ser un puente entre la anterior serie de ficheros en disco CBM y otra serie que tratará sobre las estructuras de información en soportes magnéticos, o lo que es lo mismo, diversas estructuras de organización de los ficheros en las memorias de masa. El presente artículo es eminentemente práctico, dejando los rigores académicos para la presente serie, pues es simplemente un ejemplo de aplicación de las estructuras de ficheros ya conocidas (secuencial, relativo y acceso directo) que permite crear una estructura de fichero más compleja como es un fichero secuencial indexado.

Antes de empezar con la realización práctica de este fichero, veamos sucintamente qué es un fichero secuencial indexado. Con los distintos tipos de ficheros que nos proporcionan los discos CBM, podemos acceder a un registro determinado por dos métodos distintos, inherentes cada uno al tipo de fichero que se está tratando:

1.—Acceso secuencial:

Hay que leer todos los datos desde el principio del fichero hasta que se encuentra el dato o registro deseado. Propio de los ficheros secuenciales.

2.—Acceso directo:

Se accede directamente al registro deseado (ficheros de acceso directo y relativos).

Un fichero secuencial indexado (que a partir de ahora, y para abreviar, llamaremos ISAM —INDEXED SEQUENTIAL ACCESS METODE—) se llama indexado porque en él el acceso a un determinado registro se realiza mediante una clave de acceso. Esta clave puede ser un campo del registro, una parte de un campo, o una combinación de varios campos. Esta clave de acceso, que evidentemente lleva asociada la dirección en donde se halla el registro indexado por ella, hay que tenerla guardada en algún sitio. En el caso que nos ocupa, la clave se encuentra (general-

mente) en una tabla ordenada, una detrás de otra. He aquí el motivo de que se llama secuencial, pues las claves de acceso están guardadas secuencialmente.

b) Diseño del fichero.

El ISAM que vamos a diseñar estará formado por dos ficheros distintos, el fichero índice que contiene las claves de acceso y el fichero indexado, en donde están los datos propiamente dichos. Veamos qué tipo de fichero de los que proporcionan los discos CBM se van a usar para el fichero índice y para el fichero indexado:

1. Fichero índice.

Para este fichero vamos a utilizar dos tipos de organizaciones diferentes, según el momento en que se vaya a procesar el fichero:

Acceso a los elementos del fichero índice: Para ello se va a usar una matriz bidimensional, en donde se va a guardar la clave en una dimensión y la dirección del registro asociada en la otra.

Almacenamiento en disco del fichero índice: Vamos a usar un fichero secuencial para guardar este índice en disco. La elección del fichero secuencial estriba en que solamente al principio del proceso se va a leer todo el fichero índice del disco, volcando su contenido en la matriz bidimensional, y al finalizar el proceso se va a volcar esta matriz al fichero en disco. Y el fichero más rápido en lectura y escritura es el secuencial.

2. Fichero indexado o de datos.

Este fichero tiene que permitir el acceso a un registro determinado. Los discos CBM nos proporcionan dos tipos de ficheros que cumplen este requisito, los ficheros de acceso directo y los relativos.

Para facilitar el comentario, vamos a centrarnos en primer lugar en el diseño de un ISAM con un fichero relativo como fichero indexado.



c. Diseño del programa.

El método de búsqueda de las claves en la matriz será el dicotómico, muy rápido en una tabla que esté ordenada.

Una vez fijados los principales tipos de datos a usar, vamos a ver el programa ejemplo de un ISAM con fichero relativo. Este programa se puede usar como rutina en un programa de gestión normal y contempla las opciones de mantenimiento de cualquier fichero, como son altas, bajas, modificaciones y consultas.

A continuación voy a describir el algoritmo del programa.

El programa tendrá dos cuerpos principales, el que gestionará el índice y el que recogerá los datos deseados del fichero indexado.

El programa se diseñará con un cuerpo principal de rutinas, comunes a los procesos principales de éste, que consistirán en altas, bajas, modificaciones y consultas.

El algoritmo es:

- 1) Inicialización de variables.
- 2) Rutinas principales.
- 3) Cuerpo principal.

- 3.1. Cargar el índice en la matriz.
- 3.2. Seleccionar operación.
- 3.3. Ejecutar operación (altas/bajas/modificaciones/consultas).
- 3.4. Si se desea otra operación, ir a 3.3
- 3.5. Si no, grabar índice en disco.

- 3.3.1. If operación=altas then altas
If operación=bajas then bajas
If operación=consultas then consultas.
If operación=modificaciones then modificaciones.

—Altas:

- (1) Entrar clave
Si clave="" then 3.2
Buscar clave. Guardar posición T
Si existe clave then entrar clave
Entrar resto del registro.
Pedir confirmación.
Si confirmación=no then entrar clave.
Buscar primer registro libre.
Insertar en índice clave+dirección primer lib. (posición T).
Incrementar número de registros ocupados.
Grabar registro.
Goto entrar clave, (1).

—Bajas:

- (2) Entrar clave
Si clave="" then 3.2
Buscar clave. Guardar posición T.
Si no existe then entrar clave, (2).
Leer registro.
Visualizar registro.
Pedir confirmación.
Si confirmación=no then entrar clave.

Guardar número registro.
Borrar clave del índice.
Insertar clave falsa+número registro al final del índice.
Marcar registro como dado de baja.
Go to (2).

—Consultas.

- (3) Entrar clave.
Si clave="" then 3.2
Buscar clave.
Si clave no existe then entrar clave.
Leer registro.
Visualizar registro.
Goto entrar clave, (3).

—Modificaciones.

- (4) Entrar clave.
Si clave="" then 3.2
Buscar clave.
Si clave no existe then entrar clave.
Leer registro.
Visualizar registro.
Modificar registro (excepto campo clave).
Goto entrar clave, (4).

En el próximo artículo os entregaré el listado de uno de los muchos programas que podrían corresponder al algoritmo aquí descrito. No obstante, a ver si os animáis y mientras tanto hacéis vuestro propio programa según este algoritmo. ■

MARKETCLUB

—Servicio gratuito para nuestros lectores particulares. Empresas 300 ptas., por línea.

MERCADILLO

- CBM 4.032. Intercambio programas. José Marcé Mestres. Calle Sevilla, 5. Tel. (93) 803 77 51, de 8 a 3, VILANOVA DEL CAMÍ (Barcelona). Referencia M-1.
- ACCESORIOS VIC-20: Ampliación de memoria 16K más varios programas, 13.000. Módulo de expansión para 6 cartuchos, 10.000. Cartucho lenguaje FORTH y manual, 7.000. Las tres cosas sólo por 25.000. Jaime. Tel. 245 46 56. BARCELONA. Referencia M-2.
- Desearía vender por 40.000 ptas.: VIC-20, con cartucho Super Expander, las dos partes del Curso, el "joy-stick" y un juego Indescomp, todo comprado en diciembre del 82. Fernando Martínez, calle La Roda, 39, 5º D. Tel. 23 41 82. ALBACETE. Referencia M-3.
- Tengo un PET 2001/8K y desearía tener el cassette "Monitor para lenguaje Máquina". También desearía contactar con usuarios o clubs de PET si los hay. José Manuel Cámara Mas, calle Castor, 32, bloque II, 3º, puerta 1. ALICANTE. Referencia M-4.

- Se busca MODEM para Commodore 64. Precio a convenir. José M.ª Maci. Horno de Ladilla, 4, Montblanc, Tarragona 860063. Horas de comida. Referencia M-5
- Vendo VIC-20 (Comprado en noviembre 1983), 2 cursos de basic, libros, programas, 4 cartuchos, juegos en cassette, cassette, joysticks. Equipo ideal para iniciarse en informática. Todo por 59.000 (Valor real 90.000). Llamar de 9 a 11 h. noche. Joan Sanz. Tel. 3218064. Travesera de las Corts, 295. Barcelona, 29. Referencia M-6.
- Compraría Cassette y aplicación de memoria 8K para VIC 20. Precio a convenir. Manuel Aranda (4354900) oficina; 91 (6179331) casa. Referencia M-7.

CLUBS

Como presentación, valga decirles solamente que soy distribuidor de Commodore en Palma de Mallorca y al mismo tiempo aficionado a los microordenadores.

Estoy intentando coordinar los esfuerzos de unos cuantos clientes y amigos para formar un club de VIC y 64 en Palma de Mallorca. También os agradeceré si me podéis ayudar con una reseña en la revista para que los aficionados de Palma sepan que intentamos formar un grupo de ami-

gos del VIC-20 y COMMODORE-64. RAMON P. SERNA SOLER. FOTO ESTUDIO 2. Plaza de Sombrereros, 2. PALMA DE MALLORCA. Tel. 21 31 62. Referencia C-1.

Desearía información sobre clubs en existencia y gente que quiera formar uno. AGUSTIN QUEVEDO VELASCO. OÑA, 55-4ºB. MADRID-34. TEL.: 202 94 28. Referencia C-2.

Desearía información sobre clubs en existencia y gente que quiera formar uno. PABLO NISTAL ALONSO c/EMPECINADO, 46 6ºB MOSTOLES. TEL.: 645 21 70. Referencia C-3.

Hace poco tiempo quedé establecido y reglamentado el Club de Programación Alaiz, en Pamplona. Está dirigido a estudiantes de B.U.P. y C.O.U. de esta ciudad, y el fin que tiene es la introducción en el mundo de la informática y programación BASIC. No tiene, por lo demás, ningún fin lucrativo. El club propiamente posee equipos COMMODORE 64, VIC-20 y diversos periféricos (impresora, unidad de disco y cassette). CLUB ALAIZ. Pza. Monasterio Santa Gemma, s/n. Tels.: 254480-257704 PAMPLONA. Referencia C-4.



BOLSA DE TRABAJO

PROGRAMADORES COMMODORE-64, excelente oportunidad para personas que dispongan de programas de gestión y utilidades y deseen comercializarlos en condiciones verdaderamente ventajosas (sólo programas de alta calidad). También programadores dispuestos a realizar programas a medida. Deberán ponerse en contacto con "EAF microgestión", preguntar por Francisco Arambudo, teléfono: (93) 231 95 87 de Barcelona, o escribir al apartado de correos 24143 de Barcelona. Referencia B-1.

SERVICIOS

• En Barcelona, clases de informática. **PLAZAS LIMITADAS**. Lenguaje BASIC. Prácticas con microordenador VIC-20. Prof. E. Martínez de Carvajal. Información: Tel. (93) 345 10 00. Señorita María José (mañanas) ó (93) 345 87 75.

Sr. Martínez (fuera de horas de oficina). Referencia S-1.

• Programación de ordenadores personales; organización explotación de ficheros; programas ordenadores auxiliares, para cuestiones empresariales, profesionales, administrativas, científicas. Mora Mas. Carlos III, 41. Tel.: 339 98 29. BARCELONA-28. Referencia S-2.

• Tengo programa para confección de documentos, cartas, textos, etc. Permite escribir líneas reales de impresora visualizando por pantalla y avisando al final de cada línea por timbre. Visualización del texto entero; modificaciones de línea; insertar líneas; grabación disco o cassette; lectura de datos de disco o cassette; control por pantalla del texto a rectificar; acepta; espacios, etc.; imprime normal y doble ancho; pregunta número de copias; salto automático de página en impresora; margen izquierdo en la página. Amadeo Bargay. Pza. Hospital 5-5º. Tel.: 874 41 93. Manresa. BARCELONA. Referencia S-3.

GALERIA DE SOFT

NOVEDADES

Microelectrónica y Control ha sacado 3 nuevos programas para el C-64. Introducción al BASIC, parte I; EASY CALC RESULT (20.000 pts.) y MACRO ASSEMBLER (7.500 pts.). La reseña de estos dos últimos saldrá en el próximo número.

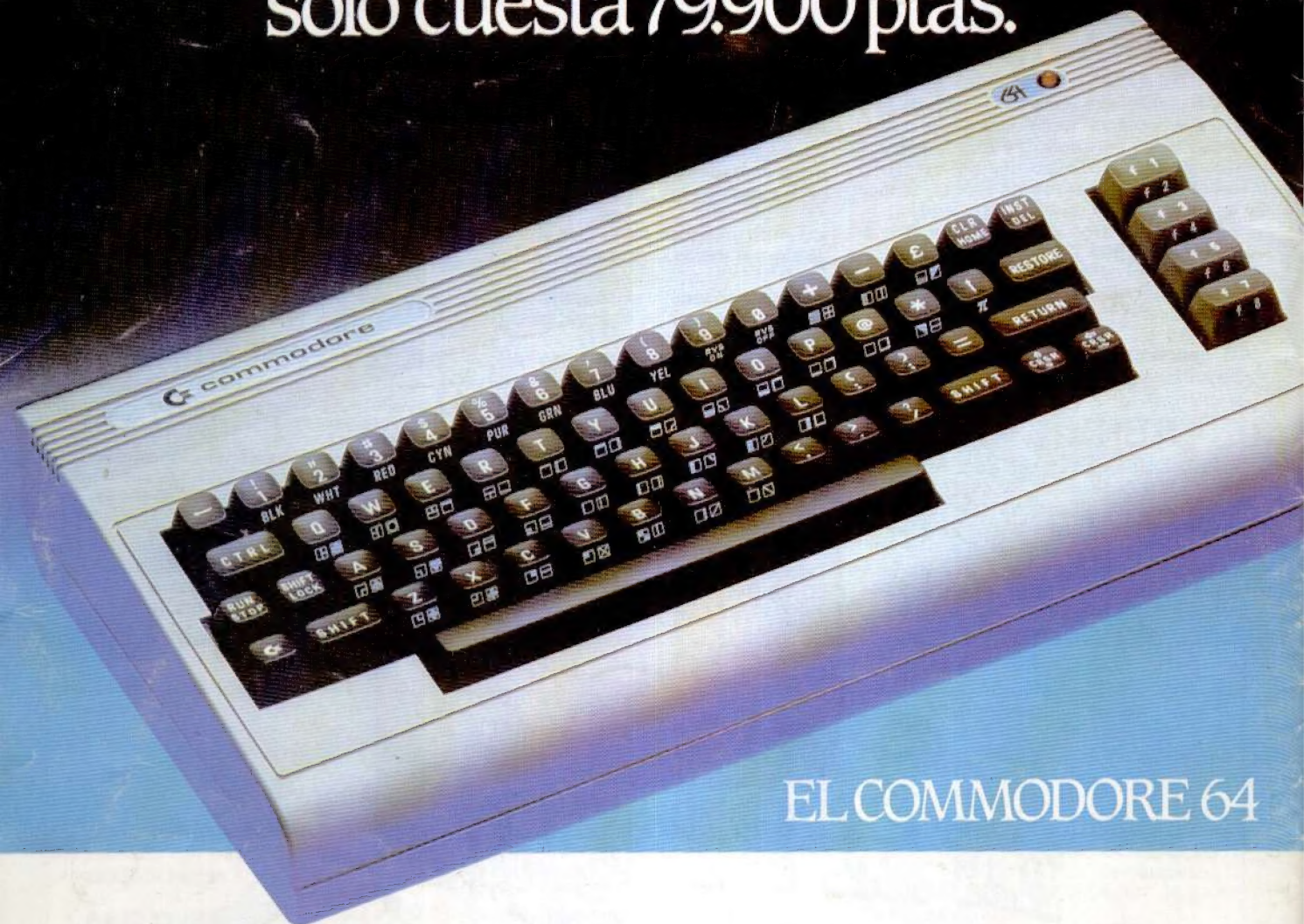
CURSO DE INTRODUCCION AL LENGUAJE BASIC, PARTE I

Con este curso no se incluyen cintas de programas, en su lugar, se ha añadido un apéndice con los listados de los programas a los que se refieren las unidades didácticas, para que el usuario los entre por sí mismo en el equipo y realice de esta manera una práctica de gran utilidad en BASIC.

Precio: 3.000 ptas.



El mejor ordenador personal del Mundo sólo cuesta 79.900 ptas.



EL COMMODORE 64

1. Capacidad total de memoria RAM de 64 K. Interpretador BASIC extendido y sistema operativo residentes en ROM.

2. Dotado del más potente chip sintetizador de sonido diseñado hasta hoy, el COMMODORE 64 ofrece 3 voces totalmente independientes con una gama de 9 octavas. El programa puede controlar la envolvente, la afinación y la forma de onda de cada voz, convirtiendo al COMMODORE 64 en el mejor simulador de instrumentos.

3. Conectable directamente a toda una gama de periféricos, incluyendo unidad de discos, impresora de matriz de puntos o de margarita, plotter, comunicaciones locales y remotas..., y mucho más.

4. Pantalla de alta resolución en color con 320 x 200 puntos directamente direccionables. Capacidad en modo carácter de 25 líneas por 40 columnas.

5. El chip de video, único en su género, permite el uso de 8 «Sprites» (figuras móviles en alta resolución y color). Los «Sprites» pueden moverse independientemente por programa de «pixel» en «pixel».

6. Teclado profesional con mayúsculas y minúsculas, más 62 caracteres gráficos, todos ellos disponibles en el teclado y visualizables en 16 colores, en forma normal o bien en video invertido.

7. Encontrará a su disposición una completa gama de programas profesionales, incluyendo proceso de textos, sistemas de información, modelos financieros, contabilidad y muchas más aplicaciones.

8. Están en fase de desarrollo otros lenguajes tales como LOGO, COMAL, PILOT, etc.

9. Opción de un segundo procesador Z-80 para trabajar con sistema operativo CP/M (R).

 **commodore**
COMPUTER

MICROELECTRONICA Y CONTROL
c/ Taquígrafo Serra, 7, 5º Barcelona-29
c/ Princesa, 47, 3º, G Madrid-8